VŠB – Technická univerzita Ostrava
Fakulta strojní
Katedra informatiky

Rozpoznání lidských akcí ve videosekvencích

Human Action Recognition in Video Sequences

2017 Jiří Pyszko

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Jiří Pyszko**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Rozpoznávání lidských činností ve videosekvencích
Human Action Recognition in Video Sequences

Jazyk vypracování: čeština

Zásady pro vypracování:

Úloha o rozpoznávání činností ve videosekvencích je velmi praktická. V současné době jsou k dispozici kamery a prostředky pro přenos nebo záznam obrazu a vzniká velké množství obrazových dat. Zatím však nejsou k dipozici dostatečně inteligentní algoritmy k jejich vyhodnocování, které je povětšinou prováděno člověkem, a stává se proto limitujícím faktorem. Cílem diplomové práce je přispět malým dílem k rozvoji této oblasti a získat zkušenost v této oblasti. V diplomové práci proveďte:

1. V rešeršní části práce stručně popište několik známých metod. Můžete se omezit na metody založené na detekci kostry.

2. Jednu z metod vyberte a naimplementujte. Metodu můžte i sám modifikovat, budete-li mít za to, že se tím zlepší úspěšnost detekce. Také reportoár činností, které nají být rozpoznávány, můžete zvolit sám. Může se jednat o činnosti vcelku elementární jako je např chůze, běh, výskoky atd.

3. Metodu řádně otestujte a dosažené výsledky pečlivě zdokumentujte. Implementaci proveďte v C/C++.

Seznam doporučené odborné literatury:

Poppe, R.: A Survey on Vision-Based Human Action Recognition, Image and Vision Computing 28 (2010) 976–99

Cheng, G., Wan, Y., Saudagar, A.N, Namuduri, K., Buckles, B.P.: Advances in Human Action Recognition: A Survey, Computer Vision and Pattern Recognition, 2015

Liang, B., Zheng, L.: A Survey on Human Action Recognition Using Depth Sensors, In Proc. DICTA 2015, pp 1-8

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Dr. Ing. Eduard Sojka**

Datum zadání: 01.09.2016

Datum odevzdání: 28.04.2017

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 28. dubna 2017 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 28. dubna 2017 .....

V první řadě bych rád poděkoval svému vedoucímu doc. Dr. Ing. Eduardu Sojkovi za jeho rady při vytváření této práce. Dále bych rád poděkoval své rodině, která mě podpořila a bez níchž bych tato práce nevznikla.

Abstrakt

Tato diplomová práce se zabývá rozpoznáváním lidských akcí ve videosekvencích. Celkem jsou navrženy tři různé algoritmy, které plní funkci rozpoznávání lidských akcí. První metoda, založená na metodě DTW, slouží pro rozpoznávání sekvencí na níchž je znám počátak a konec. Další dvě metody jsou vytvořeny s využitím neuronových sítí a zaměřují se na rozpoznávání akcí v reálném čase. Pro tyto metody jsem vytvořil testovací aplikaci programovanou v jazyce C++ pomocí knihovny OpenCV, srovnal jsem výsledky a zdůvodnil jsem jejich výhody a nevýhody.

Klíčová slova: rozpoznávání lidských akcí, neuronové sítě, Kinect, OpenCV, hloubkové mapy

Abstract

This master thesis is focused on recognizing human actions in videosequences. Overall, there are proposed three different methods for recognition of human actions. The first method, based on the DTW algorithm, serves only for recognizing actions where is begin and end known. Another two methods are created using neural networks and focus on recognizing human actions in real time. For these methods, I created a test application programmed using C++ language and OpenCV open source library. Then I compared the results and explained advantages and disadvantages particular methods.

Key Words: human action recognition, neural networks, Kinect, OpenCV, depth maps

Obsah

Seznam použitých zkratek a symbolů

DTW	–	Dynamic Time Warping
BoF	–	Bag of features
BoW	–	Bag of words
SVM	–	Support vector machine
MoCap	–	Motion capture
HOJ3D	–	Histogram of 3D joints
LDA	–	Linear discriminant analysis
HMM	–	Hidden markov model
HOG	–	Histograms of oriented gradients
ROP	–	Random occupancy pattern
IR	–	Infrared
MEI	–	Motion energy images
MHI	–	Motion history images
CGI	–	Computer generated imagery

Seznam obrázků

Seznam tabulek

1 Úvod

Rozpoznávání lidských akcí je zajímavý a široce studovaný okruh počítačového vidění, ve kterém se provádí výzkum již více než dvě desetiletí. Jeho potenciál můžeme vidět v sledovacích systémech, analýze videí, robotice nebo například v nových způsobech použití a ovládání počítačů. V době psaní diplomové práce byl největší rozsah výzkumu v oblasti rozpoznávání akcí prováděn na videosekvencích nahraných pomocí dvou kamer. Jedna snímá viditelné spektrum světla a druhá snímá hloubku obrazu.

Tato práce se zaměřuje na rozpoznávání akcí pomocí lidské kostry. Pro získání kostry se používá speciální kamera s hloubkovým senzorem, která pomocí dnes již známých algoritmů dokáže získat klíčové body kostry natáčeného člověka. Jelikož cílem práce není získávání kostry člověka, ale rozpoznávání pohybů, jsou použity již nalezené kostry ve videosekvencích.

Práce je rozdělena do několika částí. První část, ve které se právě nacházíme, seznamuje čtenáře zběžně o tom, co se dozví na stranách následujících. V druhé části práce se podíváme na technologie, které v době psaní této práce pomáhají s rozpoznáváním lidských akcí. Ve třetí části si rozebereme postup, jak lze reprezentovat pohyb pomocí zjištěných bodů kostry na jednotlivých snímcích sekvence. Čtvrtá část se zabývá rozpoznáváním lidských akcí na sekvencích, kde je předem znám počátek i konec vykonávané akce. Porovnává sekvence pomocí dynamic time warpingu (DTW) a je to první způsob, který jsem realizoval. Rozebereme si výhody i nevýhody, jeho použití. V páté části si rozebereme další způsoby, které dokáží rozpoznávat akce v reálném čase, které pracují za použití neuronových sítí. Ukážeme si 2 různé řešení a srovnáme jejich klady a zápory. Šestá kapitola je věnována experimentům, kdy dopodrobna otestujeme rozpoznávací schopnosti jednotlivých algoritmů a zdůvodníme si, proč je výsledná úspěšnost taková, jaká je. V poslední, sedmé sekci shrneme celkově obsah práce a navrhneme, jak by se dala práce dále rozšířit.

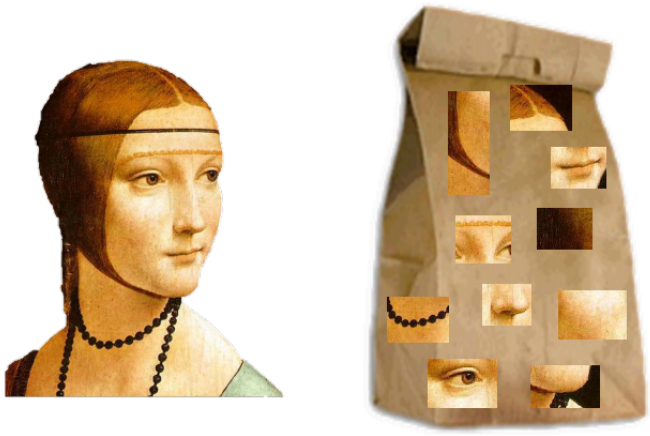
2 Současný stav

Rozpoznávání lidských akcí, přestože je to oblast zkoumána již více než dvě desetiletí, je stále náročnou oblastí výzkumu. Za prvé si musíme uvědomit, že lidské tělo je ohebné a má velkou volnost, jak se může nebo nemůže pohybovat. Navíc pro jednu akci může být nekonečně mnoho malých změn, které znesnadňují rozpoznávání těchto akcí. Druhým důvodem je diverzita lidí. Každý člověk má trochu jinou výšku, stavbu těla, a každý provádí stejný pohyb trochu jiným způsobem. Třetím problémem je to, že téměř nikdy nejsou podmínky pozorování člověka stejné - jiná pozice kamery, jiné světlo, stíny, ale problémy může způsobovat i to, že člověk nosí různé oblečení. Za těch několik desetiletí výzkumu bylo vytvořeno již mnoho metod, které se snaží přizpůsobit těmto podmínkám a z velké části eliminují tyto problémy, avšak ne všechny.

2.1 Metody s použitím rozboru videozáznamu

První pokusy v této oblasti probíhaly s použitím videozáznamu zaznamenávajícím pouze viditelné spektrum světla. Obvyklým postupem při rozpoznávání člověka v takovémto záznamu je lokalizování zájmových bodů v prostoru a času. To znamená, že každý důležitý bod má nejen svoji reprezentaci v obraze, ale i na kterém snímku byl zachycen a popřípadě pohyb tohoto klíčového bodu po jednotlivých snímcích. V kombinací s klasifikátorem SVM (Support vectotr machine) se tato metoda používá právě k rozpoznávání lidských akcí[1].

I. Laptev ve své práci [2] přidává k tomuto řešení metodu bag-of-features (BoF). V podstatě se jedná o metodu bag-of-words (BoF), která místo slov používá pro obrázky jeho vlastnosti. Na obrázku 1 je znázorněna metoda BoF.



Obrázek 1: Bag of features (převzato z [18])

Tato metoda je dobrá co se týče rozpoznávání obrázků, ale aplikování na lidské akce je náročnější.

A.F. Bobick a J.W. Davis ve své metodě představili rozšíření o motion energy images (MEI) a motion history images (MHI)[3]. Tyto metody použili na popsání dané akce člověka.

Komplexnějším postupem při řešení lidských akcí je metoda, která používá pro naučení a rozpoznání akcí 3D konvoluční neuronovou síť[5]. Metody založené jen na viditelném obrazu jsou hodně omezovány vlivem prostředí, nasvícením scény, použitím různých barev oblečení a dalšími nepravidelnostmi. Většina dnešních metod pokoušející se rozpoznat akci člověka, se zaměřuje na práci s lidskou kostrou, proto se v další sekci podíváme na tyto způsoby podrobněji.

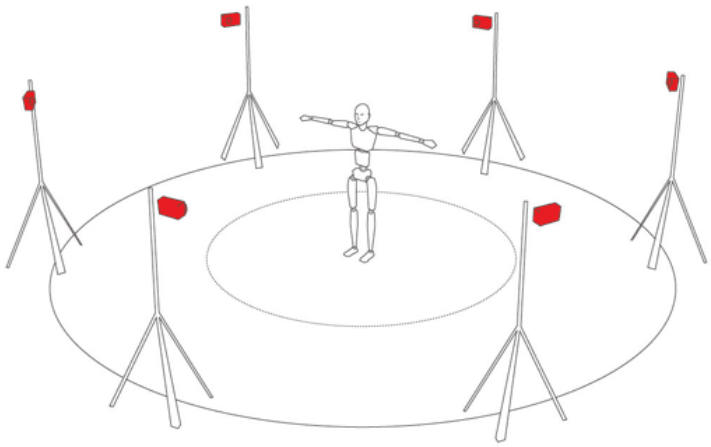
2.2 Metody pracují s hloubkovými mapami

V oblasti rozpoznávání lidských akcí je obecně považováno, že znalost bodů 3D prostoru je jen ku prospěchu. Podíváme se na několik způsobů, jak lze získat lidská kostra a následně na metody, které se v době psaní této práce používají pro rozpoznávaní lidských akcí pomocí zjištěné kostry.

Jedním ze způsobů získávání bodů v 3D prostoru je použití vícekamerového systému snímajících pohyb. Taky se tento systém nazývá MoCap(Motion Capture system).

2.2.1 MoCap

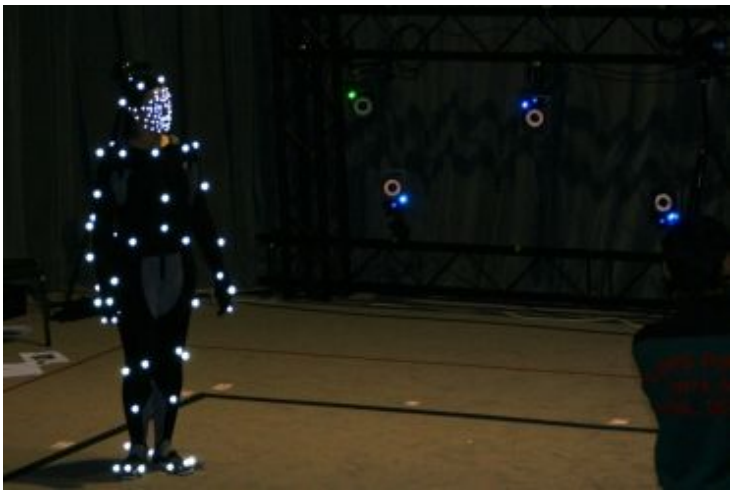
V tomto systému je subjekt snímán několika kamerami, které jsou rozmístěny na okrajích tak, že nedochází (nebo jen minimálně) k zakrytí částí těla. Na obrázku 2 můžeme vidět takové rozložení kamer.



Obrázek 2: MoCap - rozmístění kamer (převzato z [20])

V tomto systému se navíc používá speciální oblek, který má zvýrazněné některé body lidského těla. Tyto označené body lze velmi přesně snímat pomocí kamer. To zajišťuje velmi přesné rozpoznání postoje subjektu v daném čase. Navíc lze při připevnění těchto zvýrazňujících bodů na obličej získávat i to, jak se člověk tváří. Tato technologie je hojně používaná při vytváření filmů a her, kde jsou postavy generovány pomocí CGI (Computer generated imagery). Pro rozpohybování těchto postav je mnohem jednodušší a přirozenější zachytit reálného člověka dělat

daný pohyb. Daný pohyb lze aplikovat na počítačově vytvořenou postavy, než aby to animátoři vytvářeli sami. Na obrázku 3 lze vidět, jak takový oblek se zvýrazněnými body vypadá. Pro počítač jsou tyto body lehce detekovatelné a dostáváme přesný postoj člověka.



Obrázek 3: MoCap - oblek se zvýrazněnými body (převzané z [21])

Nevýhodou systému MoCap je to, že se to nedá použít při reálném rozpoznávání lidských akcí. Při nich totiž předpokládáme, že budeme mít umístěnou kameru snímající náhodné subjekty. Nebudeme po každém člověku, který projde kolem kamery chtít, aby si musel nasadit takovýto oblek. Druhým problémem je jeho cena. Vytvořit a spravovat tento systém je velmi nákladné.

Cenově přijatelnou alternativou je získávání lidské kostry pomocí zařízení, které pracuje na bázi snímání hloubky obrazu. Pomocí informace o hloubce obrazu v daném bodě lze získávat s vysokou přesností kostru snímaného člověka. Zařízení, které pracuje na tomto principu je například Kinect.

2.2.2 Kinect

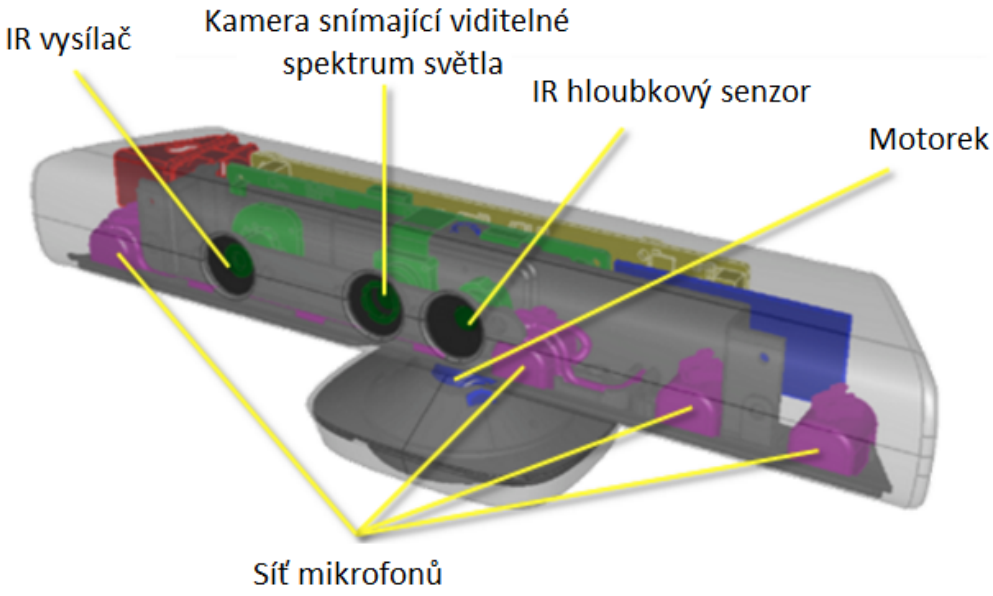
Kinect je zařízení od společnosti Microsoft vytvořené jako kamera s hloubkovým a pohybovým senzorem pro herní konzole Xbox 360 a Xbox One. Primární funkcí tohoto zařízení je umožnit uživateli interakci s konzolí bez nutnosti použití herního ovladače a hlasových příkazů. Přestože bylo toto zařízení primárně určeno pro použití s těmito konzolemi, lze toto zařízení připojit k PC.

První verze Kinectu vyšla v roce 2010 pro konzoli Xbox 360 a měla rozšířit schopnosti této konzole a nalákat nové zákazníky. Později, v roce 2012, byl vytvořen Kinect schopný propojení s PC. V roce 2013 s příchodem nové konzole Xbox One vyšla nová, vylepšená verze Kinectu (v2.0).

Přestože byl Kinect původně zamýšlen a využíván jako herní pomůcka ke konzolím, v dnešní době má hojné využití i v jiných oborech. Primárně je Kinect využíván díky svému hloubkovému senzoru. Toto umožňuje Kinectu zaznamenávat nejen klasický viditelný obraz, ale nalézt v tomto

obrazu i jak daleko jsou jednotlivé objekty vzdálené od kamery. S příchodem Kinectu dostali vývojáři do ruky jednoduché zařízení, které mohli využívat pro své aplikace, které pracovaly s výpočtem vzdálenosti objektu od kamery.

Hloubkový senzor Kinectu funguje na principu vysílání infračervené mřížky, která je pro lidské oko neviditelná. Zařízení má v sobě 2 kamery. Jedna sloužící k zaznamenávání viditelného světla a druhá, která zaznamenává infračervený obraz. Kinect pomocí infračerveného emitteru vysílá mřížku, která osvětluje (v infračerveném pásmu) dané prostředí. Po přečtení této mřížky IR senzorem dokáže Kinect určit, jak daleko se objekt od kamery nachází. Zajímavou funkcí tohoto zařízení je schopnost rozpoznávat uživatele stojícího před tímto zařízením a možnost zobrazení a zaznamenávání jeho kostry. Tato funkce je velmi užitečná a využijeme ji jako základní stavební kámen k rozpoznávávání lidských akcí v této práci.



Obrázek 4: Zařízení Kinect (převzato z [19])

Kinect ve verzi 2 přinesl několik vylepšení, které jsou přínosné při práci s hloubkovými daty. V první řadě vyměnil starou technologii snímání hloubkových dat. Nový systém dokázal měřit hloubková data 3 krát přesněji a zmenšil minimální vzdálenost. Dokázal detekovat uživatele z 1.8 metru na polovinu a dokázal zvýšit počet koster, které detekuje najednou na 6. Mezi zajímavé nové funkce patří i možnost měření srdečního tepu a nebo rozpoznání, jak se daný člověk tváří. Zvýšil se i počet rozpoznávaných bodů v kostře.

K rozmachu použití tohoto zařízení v aplikacích s potřebou hloubkového senzoru je i jeho celkem nízká cena, jednoduché použití a dobře napsaná dokumentace. Microsoft poskytuje plnou podporu tohoto zařízení k propojení s PC.

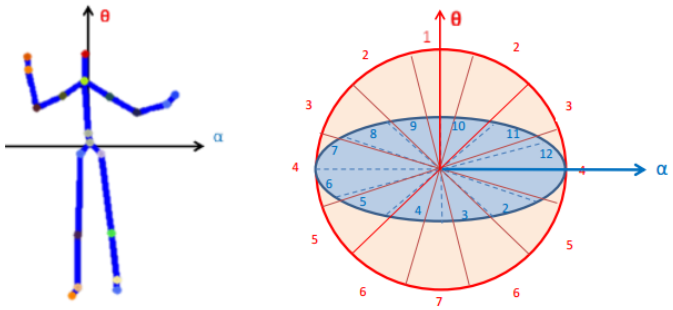
Výrobci si uvědomili, že tato zařízení mají velký úspěch a v dnešní době je možné si vybrat z mnoha různých zařízení pracující na bázi rozpoznávání hloubky obrazu. Jsou to například zařízení Structure nebo Asus Xtion Pro. Tyto zařízení mají své výhody i nevýhody. Například zařízení Structure je oproti Kinectu tak malé, že se dá používat jako přenosný senzor připevněný na iPad.

Obrázek 5: Structure Sensor pro zařízení iPad

Podíváme se na několik velmi zajímavých přístupů, které používaly zjištěnou lidskou kostru jako základní stavební kámen pro rozpoznávání lidských akcích.

2.2.3 Rozpoznávání lidských akcí použitím histogramu 3D bodů kostry

Lu Xia, Chia-Chih Chen a J.K. Aggarwal představili metodu, která používá k zaznamenání bodů v prostoru histogram 3D bodů kostry[4]. V jejich metodě se v první řadě spojí jednotlivé body kostry tak, aby vznikla lidská kostra. Pro řešení problému s postojem člověka ke kameře je v této práci zvolen způsob rotace podle osy člověka tak, jako by stál ke kameře vždy čelem. Při tomto postupu se určí pozice torza člověka a jednotlivé body se poté přepočítají tak, aby se zdálo, že stojí ke kameře čelem. Navíc je prostor kolem člověka rozdělen na 84 podprostorů, které zobecní pozici bodů kostry, aby se daly porovnávat. Na obrázku 6 je zobrazena správná rotace kostry a rozdělení prostoru.



Obrázek 6: Rozdělení prostoru kolem kostry (převzato z [4])

Nevýhodou této metody je nutnost přepočítávat každý bod podle otočení torsa. Navíc, pokud je z nějakého důvodu špatně určeno natočení torsa (špatně snímaná kostra), dochází i k výraznému posuvu jednotlivých bodů.

Na získání dominantních vlastností se používá algoritmus linear discriminant analysis (LDA). Ten je založen na třídění specifických informací, které charakterizují rozdíly mezi dvěmi specifickými třídami. Ve výsledku dokáže tento algorimtus rozpoznat jednotlivé rozdíly mezi akcemi a zvýšit tak celkovou přesnost rozpoznávání akcí.

Pro samotné rozpoznávání se používá diskrétní skrytý Markovův model (HMM). Tento model byl poprvé popsán v roce 1960. Model se hodně používá v oblasti rozpoznávání časových vzorů jako je rozpoznávání řeči, psaného slova nebo provádění gest, takže je pro tento problém ideálním kandidátem.

2.2.4 Rozpoznávání lidských akcí pomocí hloubkových pohybových map

Chen Chen, Kui Liu a Nasser Kehtarnavaz používají ve své metodě jako základ hloubkové mapy[6]. Každý snímek hloubkové video sekvence je následně promítán do tří ortogonálních kartézských ploch. Pod každou projekční maticí jsou rozdíly mezi dvěmi po sobě jdoucími promítanými hloubkovými mapami akumulovány a tím se vytvoří hloubkové pohybové mapy. Klasifikátor zvolený v této metodě funguje na principu sparse coding. Sparse coding (řídké kódování) je v době psaní práce široce využíván v oblasti rozpoznávání dat v obraze. Klasifikátor se používá například při rozpoznávání lidského obličeje. Metoda je použitá v rámci práce s neuronovými sítěmi. Hlavní idea je taková, že každá akce má silnou aktivaci na relativně malém počtu neuronů. Pro každou akci tedy jakoby existuje vlastní sada neuronů.

2.2.5 Další významné metody

Podíváme se na další možné přístupy v práci lidskými akcemi. Jednou z možností je zvýšit velikost dat o senzory pohybu, které má subjekt připnuté na těle[7]. Další možností je použít na hloubkové data dnes dobře známe deskriptory, jako je například histogram of oriented gradinent (HOG)[8]. Jiang Wang navrhl metodu, kde jsou vlastnosti získány z hloubkového obrazu pomocí metody random ocupancy patterns (RoP)[9]. Existují různé možnosti, jak lze zaznamenávat a reprezentovat body kostry. Jedním ze způsobů je metoda [10], kde jsou body kostry reprezentovány jako body ležící v Lieově grupě. Klasifikátorem v této metodě je SVM. Další metodou je rozpoznávat akce pomocí dynamic time warpingu (DTW). Tato metoda byla jedna z prvních, která mě opravdu zaujala a v této práci se jí budeme věnovat více a vytvoříme si vlastní verzi tohoto algoritmu[11].

3 Reprezentace akcí

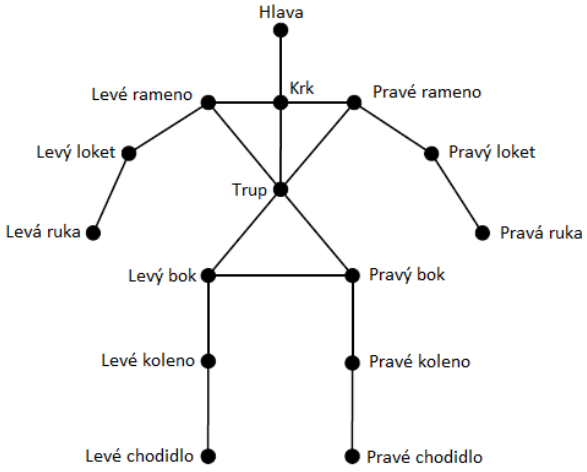
V první části zaměřující se na vypracování práce se zaměříme na to, jak budeme pracovat s informacemi, které budeme mít k dispozici. Práce se zaměřuje na způsoby rozpoznávání akcí stavějící na základech toho, že budeme mít informace o lidské kostře. Rozebereme si, jak budeme pracovat s body kostry. Jelikož nám nebudou stačit pouhé body v prostoru, vytvoříme systém, kterým propojíme jednotlivé body a následně si řekneme, jak bude reprezentována akce v této práci.

3.1 Reprezentace kostry

Lidské tělo je v této práci reprezentováno jako soustava důležitých bodů kostry, které slouží k reprezentaci postoje. Normální lidská kostra se skládá z více než dvou stovek kostí. Tento počet je v této práci redukován jen na zlomek původního počtu, protože většina spojů kostry je v rozpoznání akcí irelevantní.

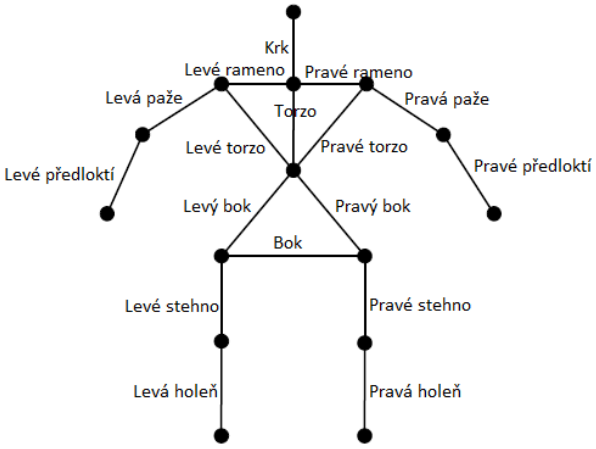
V první řadě je potřeba získat člověka v obraze. V této práci je použita metoda získávání lidské kostry. Tato kostra lze získat například pomocí zařízení Kinect, pomocí něhož lze získat důležité body kostry. Ve výsledku dokážeme získat ze zařízení Kinect 20 bodů kostry. Tyto body se nacházejí na místech, ve kterých dochází k možnému ohybu (například bod v oblasti kolena).

První věc, která lze provézt na této kostře je redukce bodů. To provádíme proto, že některé body jsou buď zbytečné pro rozpoznávání akcí, nebo jsou dva body příliš blízko u sebe, tak se jeden zanedbá. Příkladem tohoto přebytečného bodu je například bod chodidla, kde se poblíž nachází bod na pozici paty člověka, proto lze bod paty redukovat. Takto to provádíme i u dalších bodů. Po redukci počtu bodů se z původních 20 dostáváme na 15 bodů kostry. Druhým přínosem, která tato redukce má, je zrychlení rozpoznávacího algoritmu. Čím méně bodů budeme muset zpracovávat a porovnávat, tím rychleji dostaneme výsledek.



Obrázek 7: Důležité body kostry

Druhým krokem je zapracování těchto bodů do větších celků, jelikož zatím jsou to jen body v prostoru, které moc lidskou kostru nepřipomínají. Proto je použito propojování těchto bodů. Tyto body se propojí tak, že ve výsledku vytvoří model lidské kostry (například propojení mezi bodem v oblasti chodidla a bodem v oblasti kolena). Propojením těchto bodů dostaneme 17 lidských částí. Ve výsledku toto propojení již připomíná kostru.



Obrázek 8: Důležité části kostry

3.2 Zpracování kostry

Poté, co jsme získali jednotlivé důležité části kostry, je musíme zpracovat, abychom je mohli použít v rozpoznávacím algoritmu. Jedním ze způsobů je otáčet kostru podle toho, jak je daný pozorovaný člověk natočen ke kameře. Všechny body by se pak musely přepočítávat podle tohoto způsobu. Má to několik nevýhod. V první řadě samotné přepočítávání je časově náročné. I když se kostra otočí, tak tyto body nejsou přesně na stejných místech v obraze, jako to je u jiného subjektu se stejnou akcí. Tato nepřesnost se dá částečně řešit rozdělením okolí středu kostry do kvadrantů, a pak zjišťovat, které body do jakého kvadrantu patří.

V této práci se ovšem používá jiná metoda, která pracuje právě s lidkými částmi. Počítáme úhly mezi jednotlivými částmi kostry. Tento způsob nevyžaduje otáčení kostry, jelikož všechny tělesné části jsou otočeny stejným způsobem. Při 17 částech si můžeme spočítat, že veškeré úhly mezi všemi částmi spočítáme dle vyorce: 17*16/2 = 136 úhlů. Tyto úhly reprezentují postoj člověka v obraze.

Samotná akce je pak reprezentována jako posloupnost těchto zjištěných postojů člověka na jednotlivých snímcích záznamu.

V další části se podíváme na samotné metody, které lze použít pro klasifikaci a rozpoznání akcí.

4 Rozpoznáváni akcí v uložené videosekvenci

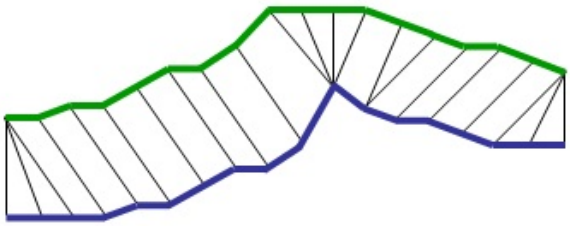
V této kapitole si ukážeme, jak lze rozpoznávat akce ve videosekvencích. Popíšeme si způsob, který rychle dokáže zpracovat a uchovat jednotlivé pohyby, ale při rozpoznávání akcí dochází k velkým prodlevám. Uvedeme si několik vylepšení, které jsem aplikoval a podrobně si popíšeme výhody i nevýhody tohoto způsobu.

Rozpoznávání akcí čelí několika překážkám. V první řadě si musíme uvědomit, že každá akce, i když stejná, může být prováděna trochu jiným pohybem a trochu jinou rychlostí. Proto je těžké pro klasifikátory určit, který pohyb sleduje.

4.1 Metoda dynamic time warping

První metoda, kterou lze použí pro rozpoznávání akcí je Dynamic time warping (DTW). Tato metoda se jevila jako vhodným postupem pro řešení problému, jelikož dokáže rozpoznávat akce, kdy lidé dělají stejný pohyb každý jinou rychlostí. Při vytváření a testování ovšem vyjdou najevo jeho výrazné nedokonalosti, jako je pomalá rozpoznávací rychlost a nutnost předem znát velikost daných pohybů.

DTW je dobře známý algoritmus na porovnávání dvou sekvencí, ve kterém se každá sekvence provádí jinou rychlostí. Sekvece jsou pomocí DTW porovnávány a cílem je najít co nejlepší schodu sekvencí. Původně byl algoritmus DTW používán na rozpoznávání řeči, ale v dnešní době se tento algoritmus již používá ve více oblastech, jako je například i rozpoznávání lidských akcí.



Obrázek 9: Podobnost mezi dvěmi sekvencemi (převzato z [24])

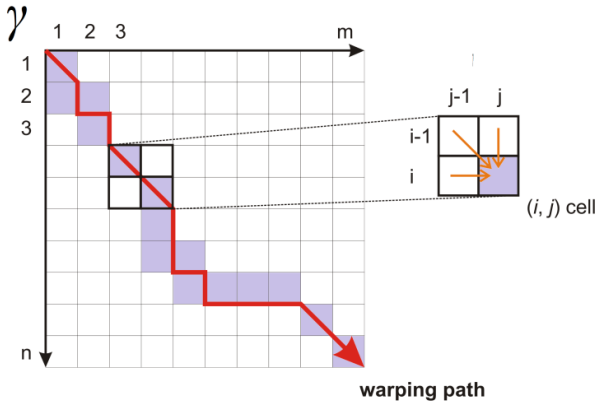
Cílem DTW je porovnání dvou časově závislých sekvencí $X = (x_1, x_2, ..., x_N)$ o velikosti N snímků a $Y = (y_1, y_2, ..., y_M)$ o velikosti M snímků.

Prvním krokem při porovnávání dvou sekvencí pomocí DTW je vytvořit si novou matici W o velikosti MxN, kde každý element matice W je spočítán podle vzorce 1.

$$f(i,j) = d(q_i, c_j) + min\{f(i-1, j-1), f(i-1, j), y(i, j-1)\} \tag{1}$$

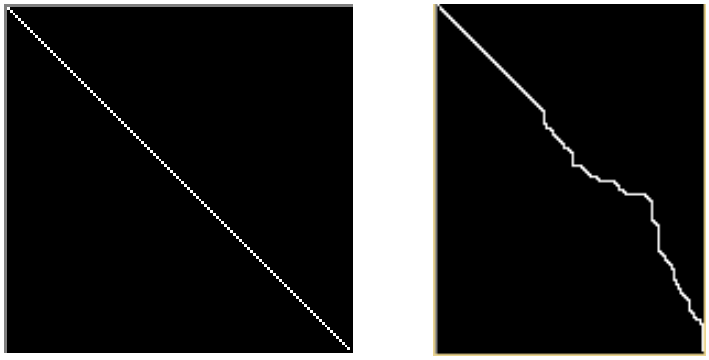
Ve vzorci 1 jsou indexy i a j pozice ve W matici. $d(q_i, c_j)$ je rozdíl úhlů v mezi 2 snímky, mezi kterými se porovnávají úhly částí kostry. Poslední částí je vyhledávání nejmenšího předcházejícího prvku. Ve výsledku to znamená, že vždy k dalšímu kroku cesty je vybrána nejmenší

předcházející cesta a je k ní připočítán rozdíl mezi snímky. Čím jsou sekvence podobnější, tím menší hodnota cesty nám vznikne. Pokud jsou sekvence úplně stejné, tak dostáváme cestu o hodnotě 0. Tím jsme dostali výslednou matici.



Obrázek 10: Postup výpočtu DTW (převzato z [25])

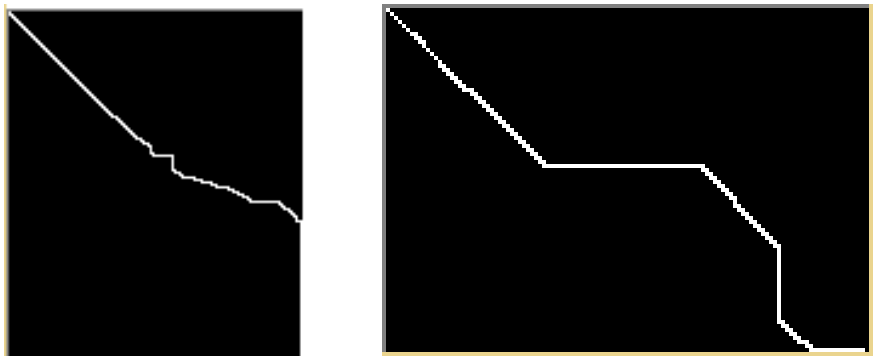
Na obrázcích 11 a 12 můžeme vidět DTW matice porovnávaných sekvencí. Na obrázku 11a je znázorněná DTW matice, která vznikne porovnáváním 2 úplně stejných sekvencí. Hodnota cesty v tomto případě je rovná 0. Na obrázku 11b je znázorněna DTW matice, která vznikla srovnáváním dvou podobných sekvencí prováděných každá jinou rychlostí.



(a) Stejná data (b) Podobná data

Obrázek 11: Podobné DTW matice

Další příklady jsou mezi různými sekvencemi. V těchto případech můžeme vidět, že algoritmus jen těžko hledá korespondující postoje postavy. Znázorněný graf je oproti podobným sekvencím více deformovaný a výsledná hodnota cesty je oproti podobným sekvencím řádově vyšší a výsledek je vyhodnocen tak, že se jedná o úplně jiné pohyby.



(a) Kratší sekvence (b) Delší sekvence

Obrázek 12: Rozdílné DTW matice

Abychom určili, jestli jsou si akce podobné pomocí těchto matic, snažíme se najít nejlepší cestu. Čím byly akce podobnější, tím menší hodnotu cesty najdeme. Samotné klasifikování, jestli se vstupní akce podobá nějaké akci uložené v klasifikátoru probíhá tak, že se srovnává vstupní akce se všemi akcemi v klasifikátoru. Podobná akce pak bude ta, která má nejmenší hodnotu cesty. Algoritmus následně vyhodnotí tuto akci jako tu, která se prováděla na videosekvenci[25].

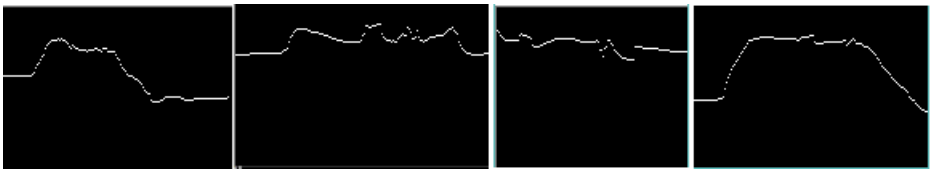
Rozpoznávání pomocí DTW má své výhody i nevýhody. Výhodou tohoto algoritmu, že na jeho naučení je potřeba jen minimální čas. Nevýhodou tohoto algoritmu je jeho časová náročnost na rozpoznávání. Je to důsledek toho, že klasifikátor prochází všechny údaje uložené v databázi a porovnává pozorovaný pohyb s uloženými akcemi. Druhou nevýhodou související s první je to, že uložené pohyby zabírají velké množství dat na disku.

4.2 Metody urychlení DTW

Jelikož při větším počtu dat použitých pro porovnávání dochází k velmi pomalému výpočtu DTW matice, musíme použít nějaký způsob, kterým bychom dokázali urychlit běh programu. První možností, která se nabízí, je redukovat počet vstupních úhlů.

Když se podíváme na běžnou akci, kterou člověk vykonává, tak nepoužívá k vykonání té konkrétní akce všechny své tělesné části. Vezměme si pro příklad člověka, který mává. Všimneme si, že člověk při této akci používá jen části jedné ruky a zbytek těla zůstává v klidu. Akce si tímto můžeme rozdělit na jednoduché akce, kde člověk používá jen pár částí těla a zbytek zůstává v klidu nebo jsou pro danou akci irelevantní, a na akce komplexní, kde člověk využívá k vyjádření akce všechny své tělesné části.

Na obrázku 13 je zobrazen průběh pohybu ruky při akci pití. Na x ose jsou jednotlivé obrázky videosekvence a na y ose je hodnota úhlu mezi rukou a torsem. Pro zobrazení byly určeny právě úhly, které mají velký rozsah změny. Na obrázcích si můžeme povšimnout, že i když se jedná stále o stejnou akci, tak rozsah i způsob pohybu se značně liší. Při redukci úhlů pak hledáme právě takové části, které mají velký rozsah pohybu.



Obrázek 13: Znázorněný rozsah úhlů mezi rukou a torsem

V této části se podíváme na dva způsoby redukce počtu úhlů, kterými se dá určit, které kombinace tělesných částí se bude používat.

4.2.1 Dynamické určení použitých úhlů

Prvním způsobem, kterým lze počet úhlů redukovat, je vzít jen ty prvky, které mají ve videosekvenci výrazný pohyb. Uveďme si příklad, že člověk ve videosekvenci mává levou rukou. Pokud si určíme, že budeme brát jen úhly, které se opravdu pohybují, tak to budou úhly mezi částmi levé ruky a ostatními částmi těla. To znamená, že nás nezajímá, jaké úhly mezi sebou svírají například části nohou.

Tento způsob má i své úskalí. Prvně musíme správně určit thresholdy (prahové hodnoty), kdy ještě považujeme rozsah pohybu za významný a kdy již významný není. Problém s nastavením správného thresholdu spočívá v tom, že člověk má při různých akcích různý rozsah pohybu. To by znamenalo, že pro každou akci se musí použít jiný threshold. Například jiný rozsah pohybů je při kopání do míče nebo například telefonování, kdy se uživatel téměř nehýbe. To můžeme získat metodou pokus-omyl, kdy budeme rozhodovat podle toho, kolik výsledných úhlů získáme. Pokud by jich bylo málo, nebude počet stačit na korektní rozpoznávání pohybu. Pokud by jich bylo naopak příliš hodně, ztratí redukce počtu úhlů smysl. Druhým problémem je to, že člověk v jedné sekvenci může používat trochu jiné částí než člověk v druhé sekvenci. Redukce chyby v tomto způsobu dosáhneme tak, že použijeme více sekvencí jednoho pohybu a určíme si jen ty úhly, které se pohybovaly na většině těchto sekvencí.

4.2.2 Předdefinované použité úhly

Druhým způsobem, jak lze počet úhlů redukovat, je vytvořit tabulku tělesných částí, které se opravdu pohybuj. Tu vytvoříme pozorováním člověka provádějícího danou akci a vyhodnocením důležitých částí pohybu. V této práci jsem pracoval jen s dynamickým určením a předdefinované tělesné části nebyly použity.

I přes tyto optimalizace je metoda stále příliš pomalá. Pokud bychom chtěli zjistit, co daný člověk na záznamu dělá, při větším počtu rozpoznávaných akcí a při obsáhlé množině trénovacích dat bychom museli čekat příliš dlouhou dobu, než aby se tento způsob dal reálně někde použít. Proto se zaměříme na způsob, který by mohl pracovat v reálném čase. Jedním ze způsobů je použitím neuronových sítí.

5 Rozpoznávání akcí v reálném čase s použitím neuronových sítí

Jelikož chceme, aby naše algoritmy dokázaly pracovat na snímaných záznamech v reálném čase, zaměřil jsem se na vytvoření algorimů, které to dokážou. Zajímavým řešením tohoto problému je použití neuronových sítí. Jejím vzorem je struktura lidského mozku a v době psaní této práce se široce používají na rozpoznávání obrazů a zvuků.

V této sekci se podíváme na dvě vybrané metody, které lze realizovat při rozpoznávání akcí v reálném čase pomocí neuronových sítí. Obě metody se nejprve naučí trénovací množinou a poté se testuje jejich úspěšnost s použitím testovacích dat. Neuronové sítě jsou velmi zajímavým postupem v řešení rozpoznávání lidských akcí. Oproti první metodě, která se zaměřuje na to, že se dokáže rychle naučit, jak daná akce vypadá, ale jejich poznávání je velmi pomalé, přináší neuronové sítě možnost, jak poznávat akce v reálném čase.

5.1 Neuronové sítě

Počítačoví vědci jsou již dlouhá léta inspirováni fungováním lidského mozku. V roce 1943, neurolog Warren S. McCulloch a Walter Pitts vytvořili první model umělé neuronové sítě. Ve své práci popisují koncept neuronu, buňky náležící do sítě buněk, které dostávají vstupní data, zpracovávají je a vygenerovávají výstup. Na tento koncept navázalo velké množství vědců a výzkumníků. Samotná neuronová síť se nesnaží přesně kopírovat fungování lidského mozku, ale je vytvořená tak, aby počítač dokázal pomocí této neuronové sítě vyřešit daný problém[15].

Některé problémy jsou pro počítač lehce vyřešitelné, ale člověk s nimi má problémy. Příkladem může být počítání s vysokými čísly, kdy počítač dokáže okamžitě, bez vynaložení většího výkonu spočítat výsledek. To se ovšem nedá říct o většině lidí. Z druhé strany existují problémy, které dokáže člověk vyřešit okamžitě, ale počítač je nedokáže. Pokud ukážeme dítěti fotku se zvířátky, dokáže říct které je které. Počítač ovšem tak jednoduše rozpoznávat věci nedokáže. Na některé z těchto těžko pro počítač vyřešitelných problémů můžeme používat právě neuronové sítě [15].

Ani neuronové sítě se ovšem nedokážou přesně chovat tak, jako jejich lidské protějšky. Vezměme si příklad čtení čísel. Mějme napsanou sekvenci číslic zobrazených na obrázku 14.

Obrázek 14: Sekvence číslic (převzato z [15])

Většina lidí dokáže bez sebemenšího zaváhání přečíst tyto číslice jako 504192. Nám se sice může zdát, že to je jednoduché, ale v každé hemisféře našeho mozku se nachází centrum zraku obsahující 140 miliónů neuronů s desítkami miliard spojení mezi nimi. K nim se přidávají další oblasti, které pomáhají komplexně zpracovávat, co ve skutečnosti vidíme. Rozpoznávání ve sku-

tečnosti není tedy tak jednoduché. To spíše člověk je neuvěřitelně výjimečný v rozpoznávaní, na co se dívá.

Pro lidský mozek se tato úloha může zdát jednoduchá, ale jakmile se pokusíme vytvořit počítačový program, který dokáže rozpoznávat číslice, narážíme na problémy a najednou už to tak jednoduché není. Například člověk rozeznává číslici 9 tak, že si řekne, že má v horní části kolečko a k němu je ve spodní části přidaná nožička. Pokud tímto způsobem budeme psát počítačový program, možná to bude fungovat v některých případech, ale nedokáže do určit správně ve všech případech.

Neuronové sítě přistupují k problému jinak. Nebude určovat, jak každá číslice vypadá. Postup je takový, že vezmeme velkou množinu dat (v tomto případě číslic) a neuronovou síť jimi naučíme. Těmto datům se také říká trénovací množina. Čím větší množinu poskytneme na naučení, tím přesněji bude neuronová síť pracovat, ale tím delší dobu se bude také učit. Na obrázku 15 vidíme příklad této testovací množiny pro číslice.

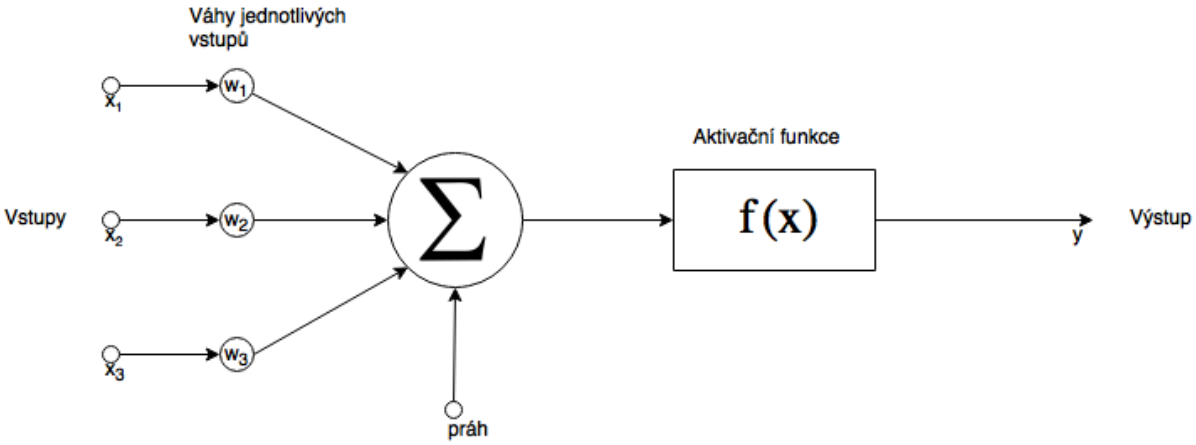
Obrázek 15: Trénovací množina (převzato z [15])

Neuronová síť použije tyto příklady v trénovací množině na to, aby si automaticky vytvořila pravidla, podle kterých dokáže rozpoznávat tyto číslice[15].

5.1.1 Neuron

Jak už název napovídá, neuronová síť se skládá z jednotlivých neuronů. Je to základní stavební kámen, který má mnoho vstupů, ale je jeden výstup. Neuron dokáže pracovat ve 2 módech:

trénovací a používací mód. V trénovacím módu se neuron učí, jak se má chovat ke vstupům, abychom dostali správný výstup.



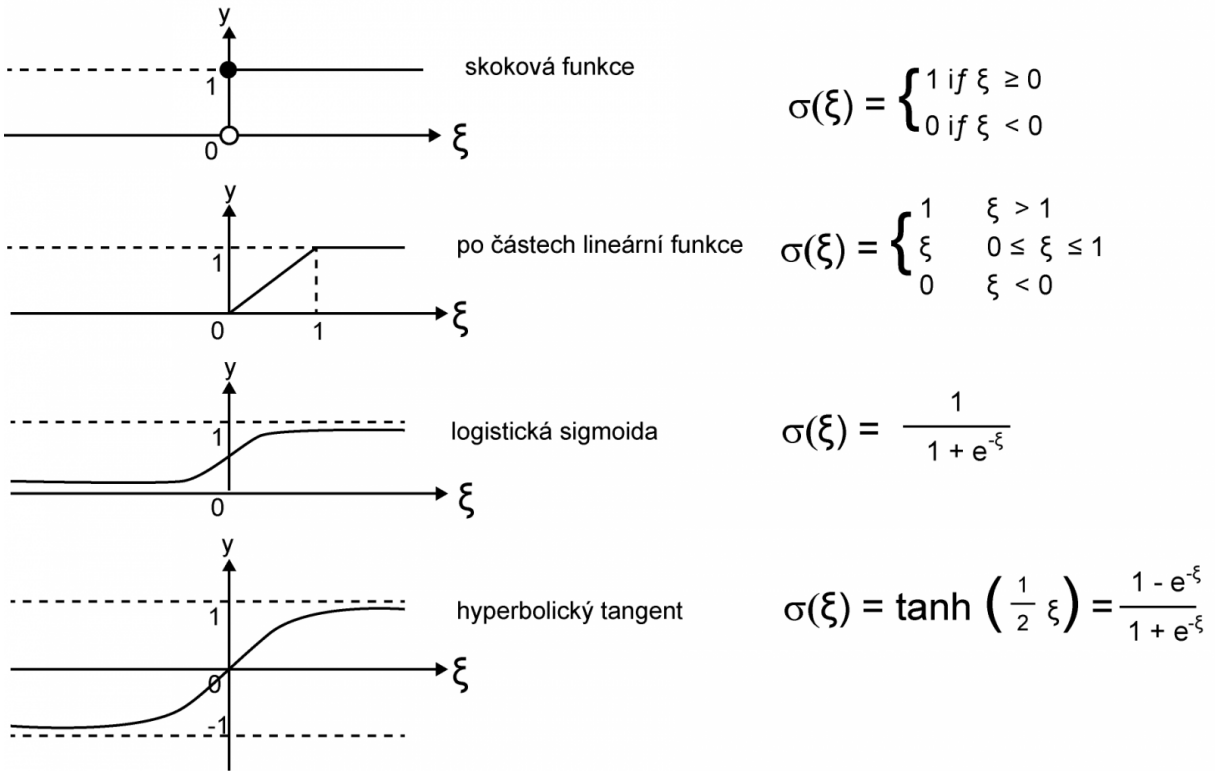
Obrázek 16: Schéma neuronu

Existuje mnoho typů neuronů. Od těch nejjednodušších, až po složité popisující chování skutečného neuronu. Na obrázku 16 je znázorněno schéma jednoduchého neuronu. Ten obsahuje několik vstupů. Každý vstup je vážen, aby se určilo, jak je daný vstup pro daný neuron důležitý. Tyto váhy se určují v době, když se neuronová síť učí. Čím je hodnota váhy pro daný vstup větší, tím je daný vstup důležitější. Vážené vstupy se poté sečtou. Sečtená hodnota pokračuje do bloku s aktivační funkcí. Aktivační funkce je prvek neuronu, který rozhoduje, zda a jaký výstup bude mít daný neuron. Matematicky lze vyjádřit neuron jako:

$$y = f(\sum_{i=1}^{N} (w_i x_i)) \tag{2}$$

V této rovnici je y myšlena výstupní hodnota. Provádí se suma jednotlivých vážených funkcí a nakonec se podle hodnoty výsledné sumy rozhodne pomocí aktivační fukce, jaký výstup bude daný neuron mít.

Aktivační funkce slouží na transformování dat, které vznikly sumou vstupů na správný výstup. V této práci je jako aktivační funkce použit symetrický sigmoid, kde výsledky jdoucí z neuronu mohou nabývat hodnot v rozsahu od -1 po 1. Na obrázku 17 jsou ukázány nejběžnější aktivační funkce, které s v době psaní této práce používají.



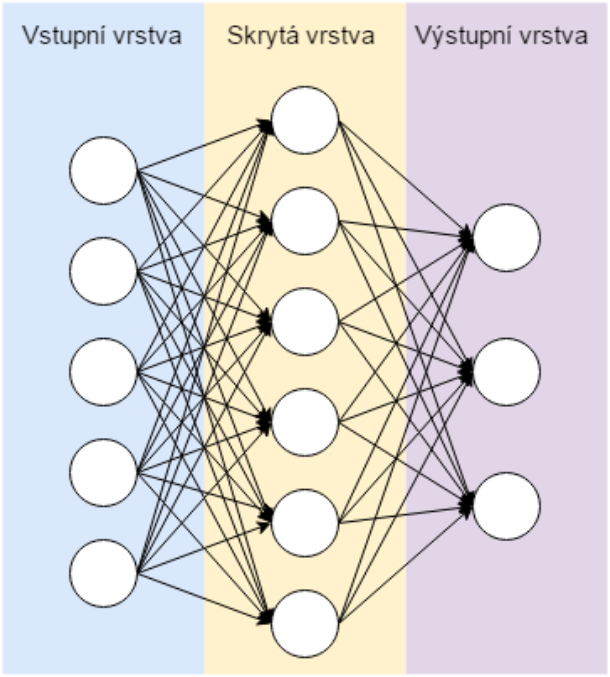
Obrázek 17: Aktivační funkce (převzato z [22])

5.1.2 Struktura neuronové sítě

Neuronová síť se skládá z velkého množství těchto neuronů. Nejběžnější model neuronové sítě je rozdělen na 3 vrstvy, kde každý neuron v jedné vrstvě je propojen ke každému neuronu vrstvy následující. Propojení těchto neuronů se nazývá synapse. Počet synapsí určuje velikost či mohutnost neuronové sítě. Čím je počet synapsí v síti větší, tím větší je počet dat, která dokáže uchovávat. Každá synapse má svoji váhu. Ta určuje, jak důležitý je daný vstup pro daný neuron. Násobí se tak příchozí hodnoty.

- Vstupní vrstva (Input layer), která přijímá data, které chceme neuronové síti poskytnou. Počet vstupních dat by měl být stejný pro každý nový záznam.
- Skrytá vrstva (Hidden layer) se nachází mezi vstupní a výstupní vrstvou. Výstupy a vstupy do této vrstvy jsou v neuronové síti skryté. Neuronová síť nemusí obsahovat jen jednu skrytou vrtsvu, ale může mít několik skrytých vrstev za sebou, než se dostanou data k výstupní vrstvě.
- Výstupní vrstva (Output layer) slouží k získávání výstupu z celého modelu neuronové sítě.

Na obrázku 18 vidíme příklad jednoduché neuronové sítě, obsahující 5 neuronů ve vstupní vrstvě, 6 neuronů ve skryté vrstvě a nakonec 3 výstupní neurony. I při tak malém počtu neuronů můžeme vidět, že počet synapsí narůstá velmi rychle.



Obrázek 18: Zobrazení neuronové sítě

Při konstruování neuronové sítě záleží, kolik bude každá vrstva obsahovat neuronů. Vstupní a výstupní počet neuronů není problém určit. Vstupní počet neuronů je roven počtu parametrů jedné hodnoty, kterou neuronová síť má zpracovávat. Výstup je roven počtu různých výsledků, které je schopna neuronová síť poskytnout. Problém nastává v počtu neuronů ve skryté vrstvě. Mohlo by se zdát, že čím větší počet neuronů ve skryté vrstvě, tím bude rozpoznávání přesnější. Takhle to ovšem nefunguje a občas dosahujeme lepších výsledků s použitím menšího počtu skrytých neuronů. Druhou nevýhodou velkého počtu neuronů ve skryté vrstvě je zvyšující se čas nutný pro naučení této sítě. Na obrázku 18 jsme si zobrazili jednoduchou síť obsahující 5 vstupních a 6 skrytých neuronů a počet synapsí narostl na 30. V reálném případě může neuronová síť mít stovky a více vstupů, stejný počet výstupů a pokud bychom plánovali naučit neuronovou síť s tiscíci nebo více skrytých neuronů, časová náročnost by byla velmi vysoká. Existuje hodně různých návodů a způsobů, které slouží pro určení správného počtu neuronů ve skryté vrstvě. Při určování se bere v úvahu počet vstupních a výstupních neuronů, velikost trénovacích dat, náročnost funkce, která se má naučit a typ učícího algoritmu. To co získáme je jen odhad, že daný počet skrytých neuronů bude nejlepší.

5.1.3 Naučení neuronové sítě

Existují 2 hlavní způsoby, jak se dá neuronová síť naučit:

- S učitelem - algoritmus se učí podle toho jaké má vstupní a výstupní data. Nejznámější učící algoritmus je Back-Propagation. Tento způsob je použit i v této práci při učení sítí.

- Bez učitele - neuronová síť se učí systémem třídění vstupů a učí se je rozpoznávat.

V této práci používám metodu učení s učitelem, která používá algoritmus Back-Propagation. Back-Propagation metoda funguje na principu toho, že se veme vstupní hodnota, nechá se zpracovat neuronovou sítí a posléze se správná výstupní hodnota porovná s hodnotou, které jsme dostali z neuronové sítě. Podle toho se určí chyba a neuronová síť si upraví váhy jednotlivých synapsí tak, aby se výsledek vylepšil. Opakovaným trénováním pomocí trénovací množiny se síť trénuje tak dlouho, dokud se buď chyba nezmenší pod nějakou nastavenou prahovou hodnotu nebo neproběhne určitý počet iterací. Poté můžeme považovat danou neuronovou síť za naučenou a začít ji zkoušet na testovacích datech[15].

5.1.4 Normalizace dat

Před tím, než začneme neuronovou síť plnit našimi daty, je potřeba tato data upravit. Neuronová síť potřebuje, aby hodnoty na jednotlivých vstupech byli přibližně stejné. Nejideálnějším případem je data normalizovat, aby data na jednotlivých vstupech patřily do intervalu od 0 do 1 (popřípadě od -1 do 1). Vezměme si příklad, kdy neuronová síť zpracovává volební preference podle toho, kdy se člověk narodil, jaký má plat a zda je muž nebo žena. Řekněme, že pan Novák má 55 let, vydělává 50 tisíc a je to muž. Tyto data se musí upravit tak, aby neuronová síť mohla správně pracovat. Výsledné vstupy pak pro danou neuronovou síť můžou vypadat jako věk 0.55, plat 0.5 a pohlaví bude mít hodnotu 1.

Tato práce používá hodnotu úhlů mezi jednotlivými částmi a hodnoty jsou uchovávány v radiánech, takže nebyly potřeba žádné nutné úpravy, aby neuronová síť pracovat správně.

Nyní jsme popsali co je to umělá neuronová síť a na jakém principu funguje. Podíváme se na jednotlivé metody, které jsem použil na rozpoznávání lidských akcí pomocí neuronových sítí.

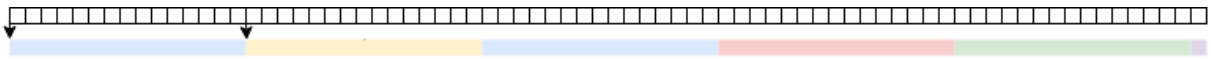
5.2 Rozpoznávání v reálném čase

Dřív, než se podíváme na metody pracující s neuronovými sítěmi, si musíme určit, jak pracovat s daty. Jak už je z nadpisu patrné, metody jsou zaměřeny na to, abychom dokázali rozpoznávat akce v reálném čase. To znamená, že sekvenci, kterou vidíme na videozáznamu musí program okamžitě zpracovat a říct nám správný výsledek. Problém nastává při přípravě dat. V první metodě DTW to probíhalo tak, že se porovnávaly dvě celé sekvence a podle toho se určilo, o jakou akci se jedná. To zde ovšem nemůžeme použít ze dvou důvodů. Zaprvé nevíme, jak jsou dané sekvence dlouhé a nemůžeme čekat tak dlouho, až se to zjistí. Druhým problémem je, že i kdybychom měli stejné sekvence jak v prvním případě a rozpoznávali jsme akci, až když ji celou provede, tak nám to nevyhovuje, protože v době, kdy dostaneme výsledek, subjekt jíž provádí akci další.

Proto je v této práci použit způsob průběžného sekvencování akcí po určitém počtu snímků. To znamená, že každá videosekvence s nějakou akcí se rozdělí po několika snímcích na jednotlivé

úseky a každý takovýto úsek bude klasifikován jako daná sekvence. Podíváme se na příklad, jak se taková sekvence může rozdělit.

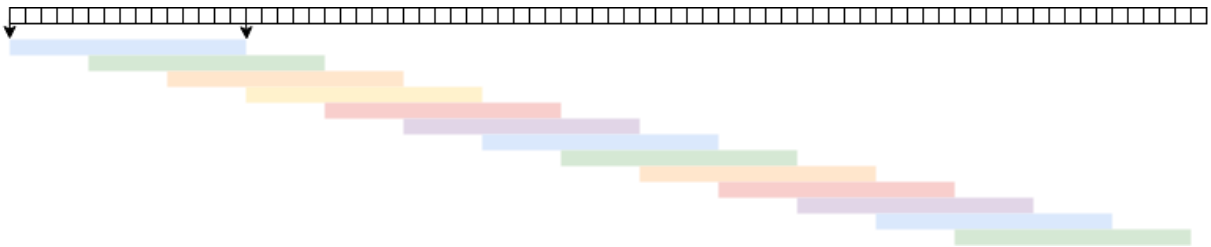
Na obrázku 19 je znázorněná taková sekvence. Ve vrchní částí jsou zobrazeny jednotlivé snímky obrazu, které jsou uloženy v trénovací množině. Každá akce v této množině je uchovávána jako sekvence těchto snímků nebo postojů kostry, podle toho, co si zrovna chceme zobrazit. Jednotlivé sekvence jsou zaznamenány rychlostí 15 snímků za sekundu. Aby se dala celá sekvence rozkouskovat, zvolili jsme úseky po jedné sekundě. To by mělo dát dostatečný prostor k tomu, aby subjekt dokázal udělat dostatečně výrazný pohyb, který by mohl být rozpoznatelný, ale umožnilo to algoritmu pracovat v reálném čase. Ve spodní části je zobrazeno, jak vypadá takto rozkouskovaná sekvence.



Obrázek 19: Rozdělení záznamů na nepřekrývající se sekvence

Způsob na obrázku 19 má velkou nevýhodu v tom, když je největší a nejvýraznější pohyb dané akce prováděn na hranách těchto rozkouskovaných sekvencí. Dochází ke ztrátě kontinuity dané akce. To by například při házení míčku bylo ve chvíli, kdyby první část pohybu házení byla na konci jedné části rozkouskované sekvence a zbytek pohybu házení by se nacházelo na první polovině sekvence následující. Je potřeba se těmto případům přizpůsobit. Proto jsem zavedl způsob segmentace, kdy se jednotlivé segmenty překrývají.

Jednou z věcí, která je potřeba určit při použití překrývané segmentace je to, o kolik se posune začátek dalšího segmentu. První možností je posouvat začátek vždy jen o 1 snímek. Tím bychom sice zajistili, že nám začátek pohybové sekvence neunikne, ale rozdíly 2 sousedních sekvencí jsou příliš malé a zbytečně by se vytvářely duplicity pro jednu akci. Tím by se zpomalil proces učení neuronové sítě, který je už tak časově náročný. Proto v této práci posouvám segmenty vždy o několik snímků. To by nám mělo optimálně zajistit, že daná akce neunikne segmentaci ale počet trénovacích dat nebude příliš obsáhlý. Na obrázku 20 je zobrazen způsob sekvencování záznamu na jednotlivé segmenty.

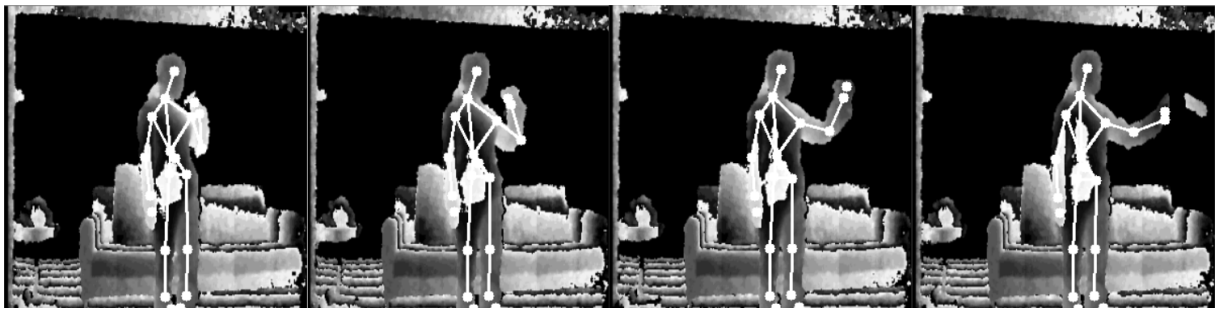


Obrázek 20: Rozdělení záznamů na překrývající se sekvence

V tomto případě je délka jednotlivých segmentů 15 snímků, což se rovná jedné sekundě záznamu. Posun jsem zvolil tak, že se začátek dalšího segmentu posune o 1/3 délky segmentu. Pro záznam akce pohybu dlouhé 10 sekund segmentované tímto způsobem dostáváme 28 segmentů.

5.3 Metoda rozdílu počátečního a koncového postoje kostry

První metoda, kterou jsem vytvořil za použití neuronových sítí pracuje na principu rozdílu počátečního a koncového postoje kostry. Pokud si vememe člověka, který provádí akci házení míčkem, jeho pohyb se skládá z pohybu jedné ruky a zbytek částí těla zůstává v klidu. Na obrázku 21 je zobrazeno, jak by takovýto pohyb mohl probíhat.



Obrázek 21: Pohyb házení

Můžeme si všimnout, že při tomto pohybu používá jen levou ruku a ostatní tělesné části jsou v klidu. Používáme sekvence, které jsme si již z vysegmentovali v předešlé části. Tyto sekvence jsou ideální ve své délce, jelikož nejsou příliš dlouhé, tak by daný úsek pohybu měl být správně zaznamenán.

Matematicky bychom mohli zaznamenat výpočet rozdílů každého úhlů je:

$$\Delta\alpha_x = \alpha_{px} - \alpha_{kx} \tag{3}$$

kde $\Delta\alpha_x$ je vypočtený rozdíl korespondujících úhlů mezi počátečním a koncovým postojem člověka, α_{px} je počáteční úhel a α_{kx} je úhel koncový. Tento výpočet provedeme pro každý úhel kostry a ve výsledku získáme 136 hodnot rozdílů úhlů mezi počátečním a koncovým pohybem.

Dalším krokem je správné vytvoření neuronové sítě pro daný problém. Počet vstupů je pevně daný (136). Každý vstupní neuron bude mít jako vstup jeden rozdíl úhlů. Počet výstupních neuronů se rovná počtu rozpoznávaných akcí. Pokud budeme například brát síť, která bude rozpoznávat 4 akce, bude mít na výstupu 4 výstupní neurony. Pokud jde o počet skrytých neuronů, v testech se jako ideální počet ukázalo, použití 100 neuronů ve skryté vrstvě neuronové sítě. Více bude popsáno v sekci testův kapile 6.

Výhodou této metody ja částečná invariance vůči postoji člověka. Když si vezmeme například házení, algoritmu je nezáleží na tom, či daný člověk při této akci stojí a nebo sedí. Toto však platí jen z části. Jelikož hodnoty úhlů se sice budou měnit jen mezi rukou a ostatním částmi těla, ale budou vznikat drobné rozdíly při akcích, kdy člověk sedí a kdy stojí. Rozdíly úhlů postojů v oblasti ostatních částech těla neprovádějících pohyb by měly být k sobě samotným v ideálním případě nulové. Nevýhodou této metody jsou opakující se, kmitavé pohyby. Pokud se pohyb provádí tak, že se postoj subjektu v dané sekvenci stihne vrátit do původního stavu

nebo uprostřed sekvence se změní pohyb a tedy na konci sekvence je úplně v jiné pozici, než je daná sekvence naučená, dochází k chybám daného postupu. Částečně je to řešeno překrývanou segmentací. Když daný segment nedokázal správně zaznamenat daný pohyb, druhý segment který ho překrýval, již tento pohyb dokázal zaznamenat správně.

5.4 Metoda s použitím průběžného postoje kostry

Druhá metoda se snaží vyřešit nedostatky první metody. I když se první způsob řešení zdál vhodný, nedokázal si poradit s jednotlivými pohyby. Jelikož se pohyb rozpoznává jako rozdíl úhlů a nebere se v úvahu ani počáteční ani koncový postoj kostry. Může docházet k tomu, že se 2 úplně rozdílné pohyby můžou zdát algoritmu podobné, i když každá akce znázorňuje úplně jiný pohyb. Stačí, aby se při nich vykonával pohyb stejnými částmi těla a algoritmus si již neví rady a dochází k mylnému vyhodnocování výsledků.

Řešením je použít takový způsob, který bere v úvahu postoj kostry. Druhá metoda se zaměřuje právě na toto řešení.

Prvním krokem je nutnost správně data zpracovat. Učení i rozpoznávání neuronových sítí je nutné provádět na malých vysegmentovaných sekvecích. Jenže taková sekvence o velikosti 15 snímků obsahuje v sobě 15 postojů člověka, který zobrazuje subjekt v obraze. Základní neuronové sítě neumožní to, že bychom na vstupy postupně vkládali těchto 15 postojů a řekli ji, že tyto postoje patří k jedné akci. Proto je použití neuronových sítí pro videozáznamy náročnější, než pro použití neuronových sítí na rozpoznávání jen v 1 obraze. Existují sice speciální neuronové sítě, které se zaměřují na práci z obrazem, ale pro tuto práci jsem zvolil jiné řešení.

V této práci jsem zpracovával data tak, aby s nimi neuronová síť dokázala pracovat jako s jedním záznamem. Dosáhneme to tím, že zvýšíme počet vstupních neuronů neuronové sítě tak, aby dokázala přečíst všechny data naráz. Výsledný počet vstupních neuronů je potřeba zvýšit podle počtu vstupů. Celkový počet vstupů je roven počtu úhlů každé kostry násobným počtem jednotlivých postojů, které neuronová síť získá. Na 15 snímkovém segmentu to bude 136*15, což nám dává 2040 vstupů. To je příliš velké množství pro jednoduchou neuronovou síť, která má poznávat jen pár různých akcí. Bylo potřeba tento počet zredukovat. Jelikož člověk provádějící pohyb neudělá obvykle velkou změnu postoje během jednoho snímků (při 15 snímcích za sekundu to odpovídá 0.067 sekundy), můžeme některé snímky ignorovat. Jelikož ale chceme větší přesnost, než měla první metoda, která nedokázala zaznamenat kmitavý pohyb, zvolíme si, že budeme číst každý pátý snímek obrazu. Ve výsledku každý pohyb je reprezentován jako průběh za určitý čas, kdy se zaznamenají jednotlivé postoje kostry.

Tímto zmenšením počtu použitých postojů dokážeme ideálně zredukovat počet vstupů neuronové sítě. Výsledná neuronová síť bude mít čtyřnásobný počet vstupních neuronů. Počet výstupních a skrytých neuronů zůstává stejný jako u první metody.

Výhodou této metody je výrazně vylepšená rozpoznávací schopnost oproti původním dvoum metodám. Nevýhodou je nutnost vytvořit obsáhlejší neuronovou síť kvůli většímu množství vstupních neuronů. Za to platíme výrazně větší časovou náročností, která je nutná pro naučení

neuronové sítě. Dalším problémem je to, že v této metodě již mnohem více záleží na tom, jestli daný subjekt při provádění akce sedí nebo stojí (například při házení míčkem). Abychom správně rozpoznali tyto akce, musíme výrazně zvýšit velikost trénovací množiny a tím dochází ještě k většímu zpomalení trénování neuronové sítě.

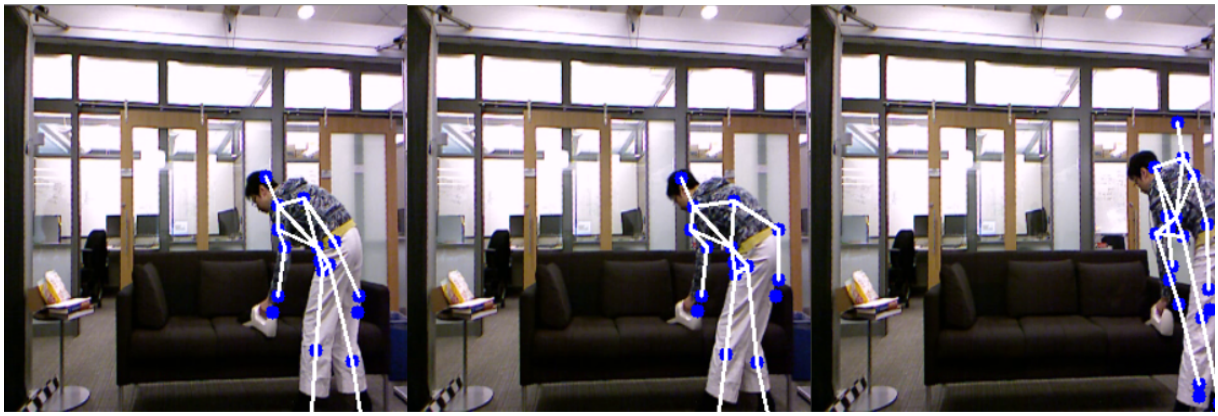
5.5 Rozpoznávání akcí pomocí neuronových sítí

Když jsme si vytvořili metody, je nutné připravit i testovaná data tak, aby je sítě dokázaly správně poznat. Prvním krokem je vytvořené neuronové sítě naučit. To se provede připravenou trénovací množinou, kde jsou data zpravována podle toho, na jaký typ sítě a metody jsou použita. Poté, co se síť naučí se může začít rozpoznávání akcí.

Testovací data se segmentují podobným způsobem, jak se to dělalo u dat trénovacích. Pro zvýšení přesnosti a počtu dat se také používá překrývaná segmentace sekvence. Jediným rozdílem při testovacích datech je to, že neposouváme sekvence o několik snímků ale vždy jen o 1. Můžeme to provést, jelikož je sice neuronová síť při učení velmi pomalá, ale pro rozpoznávání to již neplatí. Rozpoznávání můžeme provádět na každém snímku a pořád uvidíme průběžné informace, o jaký pohyb se jedná.

Jelikož nelze rozpoznat akci jen z jediného snímku, ale je potřeba mít průběžný pohyb, rozpoznává se pohyb, co provedl za poslední segment (v tomto případě sekundu). Nicméně to stále představuje rozpoznání akce v reálném čase.

Problémy nastávají ve chvíli, kdy se kostra člověka správně nerozpozná a nebo člověk provedl pohyb, na který nebyl algoritmus naučen. U nesprávného rozpoznání kostry dochází k tomu, že kostra vypadá úplně jinak, než jak člověk doopravdy stojí. Na obrázku 22 je příklad takovéto chyby kostry při akci vysávání.



Obrázek 22: Chybné rozpoznání kostry

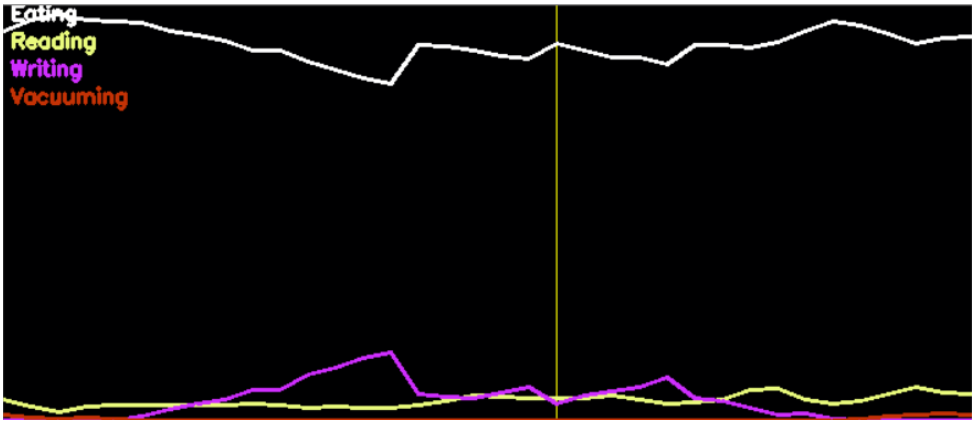
Všechny tyto nepřesnosti stěžují již tak náročný proces rozpoznávání lidských akcí.

5.6 Aproximace výsledků neuronových sítí

Z obou metod, které pracují na principu neuronových sítí dostáváme na výstupu pravděpodobnost toho, co daný subjekt na posledním segmentu snímků dělal za pohyb. V ideálních případech člověk provádí akci celou dobu bez žádných chyb kostry. V reálných případech to vždy neplatí. Může se stát, že kvůli chybě rozpoznávání kostry, nebo kvůli chvilkovému zaváhání subjektu nebo kvůli nepředvídané chybě se neuronová síť rozhodne, že se jedná o úplně jiný pohyb, než který se ve skutečnosti děje. To vytváří skokové výkyvy, při kterých je chybně určeno, o jakou akci se jedná. Jedním z řešení tohoto problému je aproximace výsledků za určitý čas.

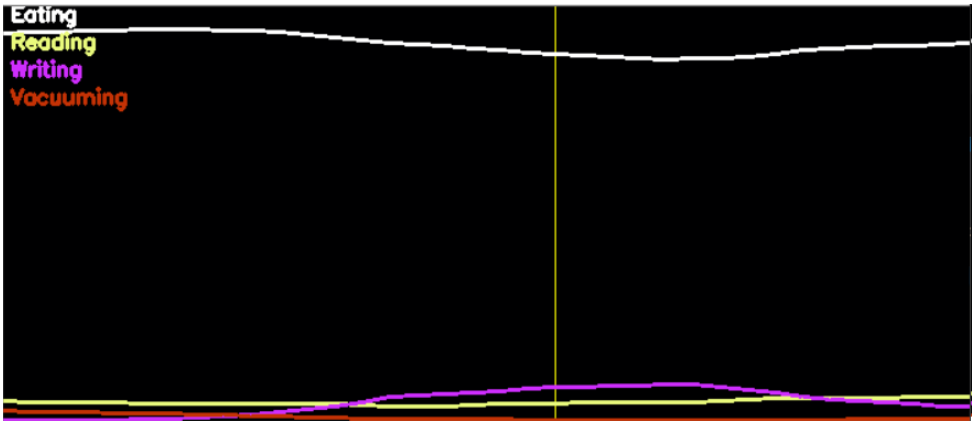
Při dosavadním přístupu jsme při rozpoznávání pohybu vždy brali jen několik snímků, které se provedly za poslední sekundu a podle toho se rozhodovalo, o jakou akci se jedná. Při aproximaci výsledků tento postup vylepšíme a pro rozhodování, o jakou akci budeme brát v úvahu výsledky rozpoznávání akcí z minulých segmentů.

Na obrázku 23 můžeme vidět příklad rozpoznávaného pohybu bez provedení aproximace. V tomto případě subjekt skutečně provádí akci jedení. Graf znázorňuje, s jakou pravděpodobností se daná akce v daném segmentu provádí. Můžeme si povšimnou, že graf je hodně kostrbatý, nachází se v něm ostré hrany, ale i přes tyto nedostatky je v celé jeho viditelné oblasti akce správně určena.



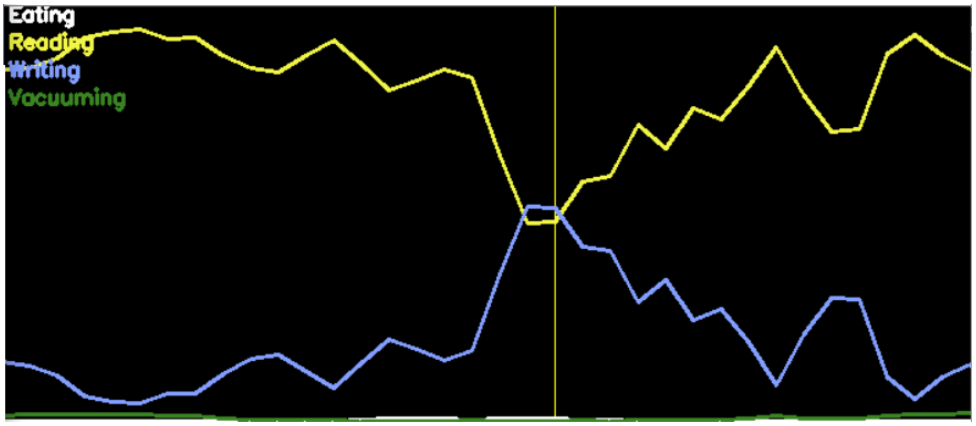
Obrázek 23: Neaproximovaný výstup sítě při jednoznačném pohybu

Pokud ovšem aplikujeme aproximační algoritmus na tato data, dostaneme celkově uhlazenější graf, který má větší rozsah mezi maximální hodnotou špatného rozpoznání a minimální hodnotou rozpoznání správného.



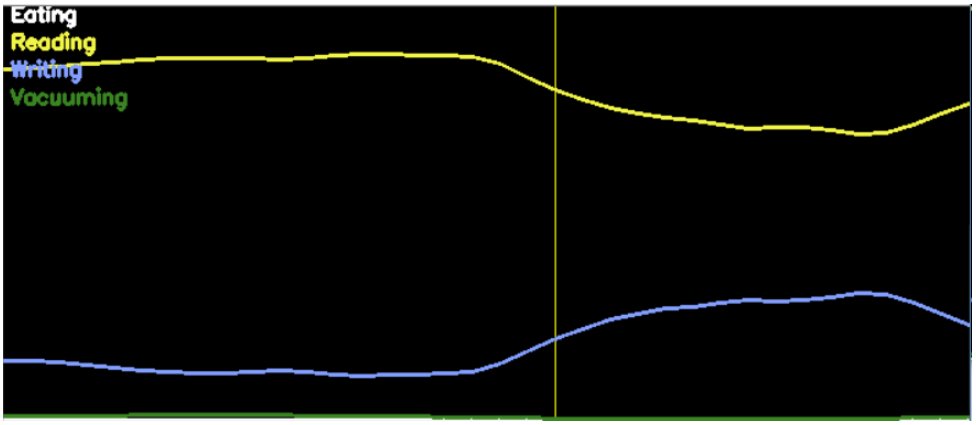
Obrázek 24: Aproximovaný výstup sítě při jednoznačném pohybu

Algoritmus nám nejen pomáhá vyhlazovat tyto nedostatky, ale dokáže tímto způsobem i redukovat počet špatně rozpoznaných segmentů. Na obrázku 25 je zobrazen takovýto případ, kdy neuronová síť v určitém segmentu špatně rozpoznala, o jakou akci se jedná. Ať už to bylo způsobeno špatnými daty nebo chybou při výpočtech, nastává v jednu chvíli moment, kdy místo akce čtení považuje algoritmus akci za psaní. Na tomto obrázku vidíme, že algoritmus v jednom bodě považuje tyto akce za téměř totožné, ale s mírnou převahou vyhodnotí akci jako psaní, což je špatný výsledek.



Obrázek 25: Neaproximovaný výstup sítě při nejednoznačném pohybu

Při použití aproximačního algoritmu na akci zobrazenou na obrázku 25 dostaneme mnohem lepší výsledek zobrazený na obrázku 26. V tomto případě jsou všechny nežádoucí výkyvy vyhlazené a rozpoznávací algoritmus rozpoznává správnou akci v celé jeho délce. Mírnou nevýhodou je to, že tyto špatné hodnoty nám mírně ovlivní další výpočty, ale jelikož tento algoritmus používáme na vyhlazení špiček, tato vlastnost nás nijak neomezuje.



Obrázek 26: Aproximovaný výstup sítě při nejednoznačném pohybu

Ve výsledku tato metoda zvýší přesnost rozpoznávaných akcí, zredukuje nepřesnosti a omezí případy, kdy byla daná akce na pár snímků rozpoznána jako úplně jiná. Nevýhodou této metody je zvýšená doba, kterou algoritmu trvá rozpoznat jiný pohyb, než se prováděl před chvílí. Při správném nastavení parametrů dostaneme vyvážený poměr mezi těmito výhodami a nevýhodami.

5.7 Shrnutí kroků algoritmu

Shrněme si, jaké jednotlivé kroky je potřeba provést, abychom získali výsledek rozpoznávání akcí při použití neuronových sítí. Prvně se podíváme na to, jaký je postup při učení neuronových sítí.

Nejprve musíme získat správná data, která dokáže program zpracovat. Pracujeme při tom s datovou sadou obsahující souřadnice bodů kostry snímaného člověka na jednotlivých snímcích obrazu. Takto dokážeme získat jak 2D souřadnici bodu v obraze, tak i 3D polohu bodu kostry v prostoru. Prvním krokem po získání těchto vstupních dat je zredukovat počet úhlů. Z původních 20 bodů zredukujeme tento počet na 16. Proč se tato redukce provádí jsem popsal v sekci zaměřující se na reprezentaci kostry. Jelikož budeme pracovat s tělesnými částmi, propojíme tyto body tak, aby připomínaly kostru člověka. Následně provádíme výpočet úhlů mezi jednotlivými částmi člověka. To nám zaručí invarianci otočení člověka vůči snímanjící kameře. Celou tuto operaci provedeme na všechny snímky záznamu obsahující danou akci a následně danou akci označíme, abychom program později věděl, o jakou akci se skutečně jedná, a uložíme. Takto si připravíme jak trénovací, tak později i testovací data.

Příprava akcí se provádí jak u trénovací, tak i u testovací sady stejně. Vyjímkou by mohlo být testování na skutečné kameře, kdyby se provádělo rozpoznávání v reálném čase a jestli se jedná o správný výsledek by musel posoudit pozorovatel.

Další postup už se liší pro trénování a testování. Při trénování se jednotlivé segmenty dané akce rozdělí podle metody překrývané segmentace. Tím dokážeme navýšit velikost trénovací

sady a připravit data pro trénování. Následně už se postup liší podle toho, kterou neuronovou síť budeme trénovat.

Při použití metody rozdílu počátečního a koncového postoje člověka vytvoříme neuronovou síť obsahující 136 vstupních neuronů, počet výstupních neuronů se určí podle počtu trénovaných akcí a počtem skrytých neuronů se více budeme zaobírat v sekci experimentů. Prozatím můžeme považovat 100 skrytých neuronů jako základní nastavení. Z celého segmentu použijeme pouze první a poslední snímek a spočítáme rozdíly korespondujících úhlů člověka. Tímto dostáváme 136 hodnot které reprezentují akci. Tyto operace provedeme na všechny trénovací sekvence. Tímto jsou data připravena a nastává samotné trénování sítě. Při této operaci se nastavují váhy jednotlivých synapsí neurovnové sítě. Jelikož je použita metoda Back-propagation, neuronová síť překontrolovává, jaké výsledky dostává při daném vstupu a váhy se přepočítávají podle toho, jak velká chyba byla na výstupu. Trénování je celkově časově zdlouhavý proces. Trénování může být ukončeno buď po určitém počtu iterací nebo pokud velikost chyby byla menší, než námi určená hranice. Jelikož docházelo při omezení počtu iterací sítě pomocí velikosti chyby k tomu, že se síť správně nenaučila, jelikož neproběhl dostatečný počet iterací, je trénování v této práci omezeno pouze počtem iterací. Naučená neuronová síť se následně uloží, abychom nemuseli provádět vždy nové trénování pro každý test.

Trénovací metoda u druhého algoritmu používajícího několik postojů kostry pro zaznamenávání a rozpoznávání akce se provádí velmi podobně. Rozdíl je v tom, které snímky segmentu používáme. Při této metodě se použije každý 5. snímek, takže celkově pracujeme se snímky 1, 5, 10 a 15. Neuronová síť vypadá také trochu jinak. Počet vstupů se zčyřnásobil, ale ostatní počty neuronů zůstávají stejné jak u předešlé metody.

Při testování zpracováváme data podobně jako u trénování. Jednotlivé sekvence rozdělíme na malé překrývané segmenty. Nyní již posouváme začátek každého dalšího segmentu o jeden snímek. Segmenty se zpracují podle zvolené metody, aby dokázala neuronová síť vstup příjmout. Načteme si naučenou neuronvou síť s nastavenýma váhama a můžeme testovat. Každý segment z testovací sady má určeno, co by měl znázorňovat, takže dokážeme porovnat výsledek z neuronové sítě se správným výsledkem. Každý výstupní neuron nám dáva výsledek, který po normalizaci znázoňuje procentuální šanci, s jakou je to právě ta daná akce.

Abychom vylepšili výsledky, zohlednili jsme vliv předešlých segmentů. Tím jsme docílili mírného zlepšení poznávacích schopností a vyhlazení skokových omylů.

V další kapitole se podíváme na testování jednotlivých algormů a zkusíme si zhodnotit, zda mají tyto algoritmy šanci uspět v tak náročné oblasti, jako je rozpoznávání lidských akcí.

6 Experiementy

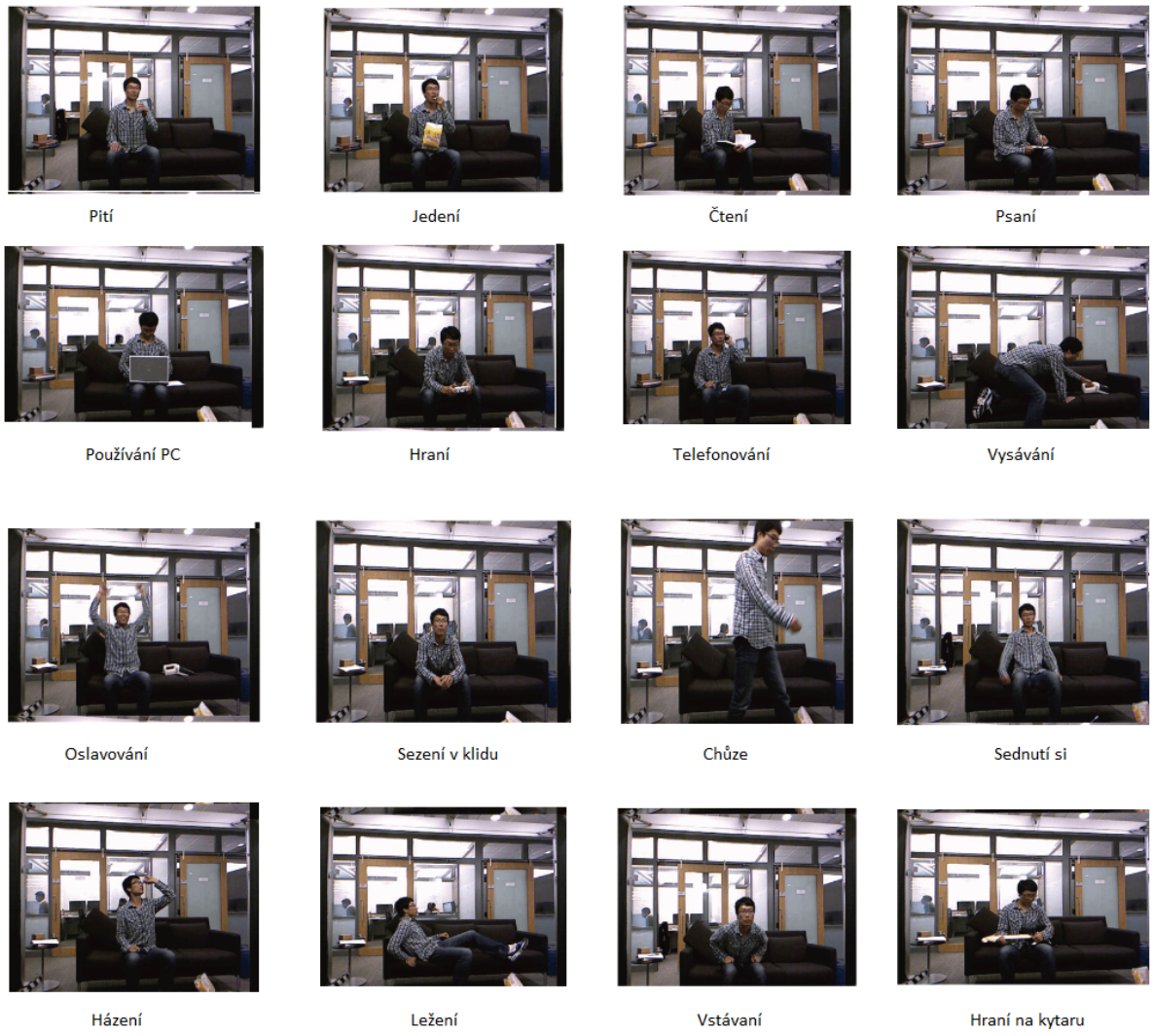
V této části si porovnáme účinnosti jednotlivých algoritmů, jejich rychlosti a správnosti rozpoznávání lidských akcí. Pro testování jsem implementoval všechny 3 algoritmy. Testovací program je psaný v jazyce C++ s použitím knihovny OpenCV. Testy jsem prováděl nad datasetem DailyActivity 3D, který zachycuje různé lidi provádějící obvyklé každodenní akce.

6.1 Prostředky použité pro testování

Prvně se podíváme na to, jaké prostředky byly použity pro vytvoření testovací aplikace. Testování jsem prováděl na dobře známem datovém balíku obsahující zaznamenané lidské akce, který se používá ve většině testování dnes vyvinutých algoritmů pro rozpoznávání lidských akcí.

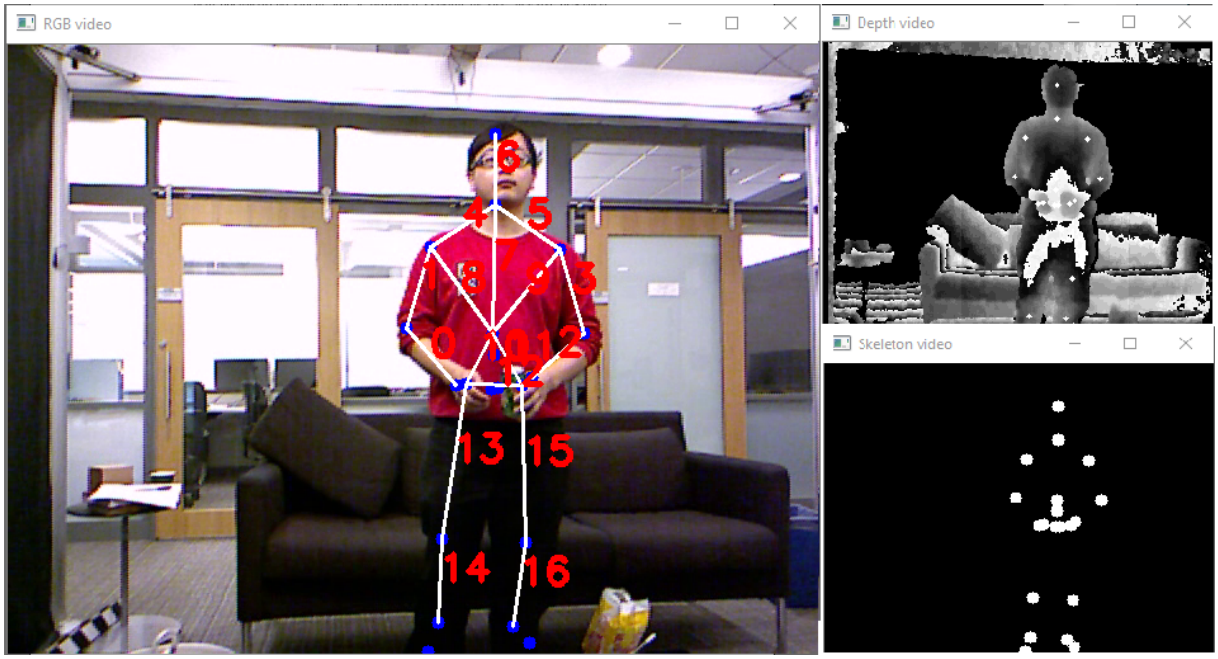
6.1.1 Použitá testovací sada

Pro trénování a testování akcí byla použitá sada MSR DailyActivity 3D dataset. MSR DailyActivity 3D dataset je balíček akcí obsahující sekvence natočené jak normální RGB kamerou, snímající viditelné spektrum světla, tak i hloubkové data a kostry nalezené zařízením Kinect. Každý bod kostry je popsán jak 3D souřadnicí, která umožňuje aplikaci rozpoznávacích algoritmů, tak i 2D souřadnicí, která určuje, kde se nachází na daném snímku obrazu. To nám dovoluje zobrazovat model kostry do probíhající videosekvence. Dataset obsahuje 16 akcí, kde každá akce je prováděna 10 různými lidmi. Akce obsažené v tomto balíčku jsou: pití, jedení, čtení knihy, telefonování, psaní, používání notebooku, vysávání, oslavování, sezení, házení, hraní her, ležení, chůze, hraní na kytaru, postavení se a sednutí si. Pokud je to možné, každý subjekt provádí akci 2 různými způsoby: jednou v sedě a jednou ve stoje. Celkový počet sekvencí nacházející se v tomto balíku je 320. Na obrázku 28 jsou ukázky jednotlivých akcí.



Obrázek 27: Různé typy akcí uložené v MSR DailyActivity 3D datasetu

Na obrázku 28 je zobrazeno, jaké informace máme k dispozici v tomto balíku. Vlevo můžeme vidět obraz zachycený kamerou snímající viditelné spektrum světla. Tento záznam má rozlišení 640x480 pixelů. V pravé horní části jsou zobrazená hloubková data. Tyto data již nejsou poskytnutá jako videozáznam a zobrazený obraz je vytvořen zpracováním těchto dat. Velikost rozlišení je oproti videozáznamu zaznamenávající viditelné spektrum světla čtvrtinový (výsledné rozlišení je pouhých 320x240 pixelů). V pravé dolní části jsou znázorněny pozice jednotlivých bodů kostry člověka. Do videozáznamu a do hloubkové mapy jsem znázornil korespondující body a části kostry, aby bylo možné porovnat, zda body kostry korespondují se subjektem zobrazeným ve videosekvenci.



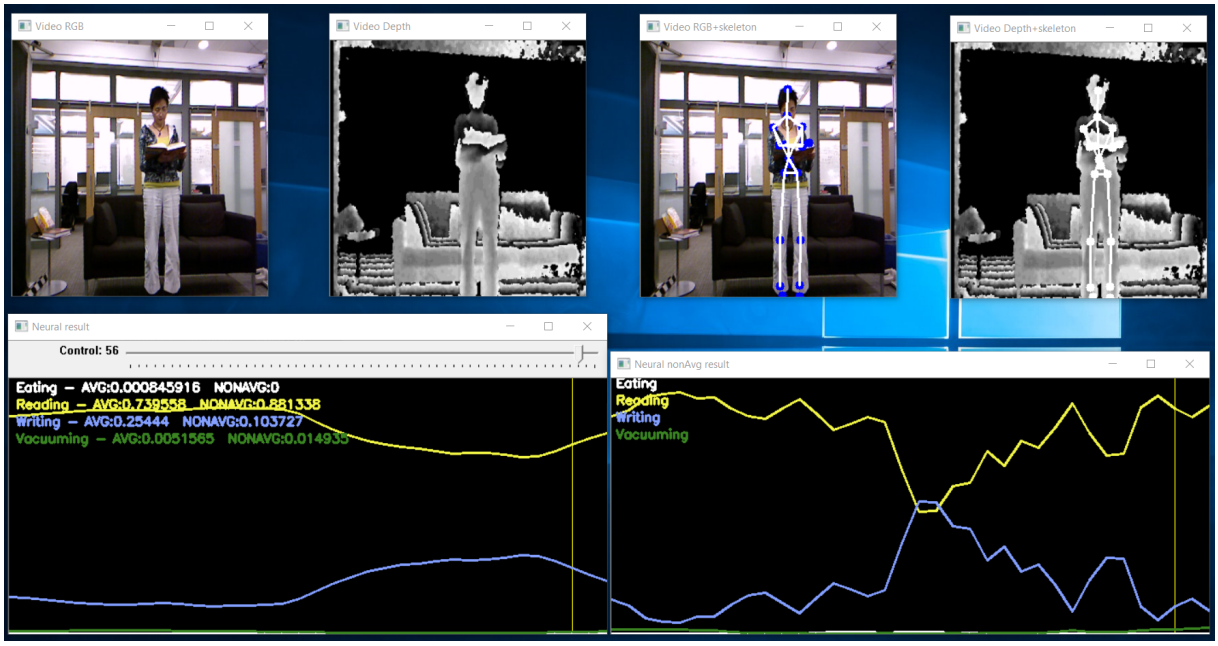
Obrázek 28: Ukázka dat z MSR DailyActivity 3D datasetu (převzané z [23])

Problém nastává ve chvíli, kdy chceme tyto akce rozpoznávat. Každá akce je prováděna jen 10 krát, a každý provádí akci trochu jinak. Některé akce jsou prováděny různými lidmi hodně odlišně. Například házení někteří lidé provádí levou rukou, jiní pravou rukou. Každý hází trochu jiným směrem. Druhým problémem těchto akcí je to, že jsou si některé příliš podobné. Například akce čtení, psaní, hraní her (subjekt při tom stojí a drží v ruce herní ovladač) jsou si hodně podobné. Subjekt při nich stojí a něco v ruce drží. Třetím problémem u těchto akcí je jejich nepřesnost. Kostry získané zařízením Kinect nejsou úplně přesné. Je to dáno tím, že člověk nelze vždy vidět celý. Jisté části lidského těla jsou skryté a Kinect jen odhaduje, kde se jednotlivé body kostry nacházejí. I viditelné body nejsou vždy správně určeny. Algoritmy nejsou dokonalé a body kostry nejsou zaznamenány přesně. To nám sice nemusí vadit u výrazně různých pohybů, ale pokud se zaměříme na již dříve zmíněné pohyby s jen velmi malým rozdílem, tak dochází k chybnému určení prováděného pohybu. Aby toho nebylo málo, samotné natočené akce nejsou dokonalé. Na videosekvencích se sice provádí daná akce, ale natočené záznamy neobsahují jen danou akci. Dochází k tomu, že subjekt na záznamu stojí před provádění akce nebo po provedení.

6.1.2 Testovací prostředí

Pro testování jednotlivých algoritmů jsem vytvořil jednoduchou konzolovou aplikaci pracující v jazyce C++. Pro všechny výstupy a zobrazovací funkce byla použita již dříve popsána knihovna OpenCV. Na obrázku 29 můžeme vidět, co všechno zobrazuje aplikace při rozpoznávání akcí v reálném čase.

Testování bylo prováděno na notebooku s procesorem Intel i5, 8GB RAM a dedikovanou grafickou kartou AMD Radeon R5 M230. Spuštěné to bylo v systému Windows.



Obrázek 29: Ukázka výstupu testovací aplikace

6.1.3 OpenCV

OpenCV je volně šiřitelná knihovna, která je používána v oblasti počítačového vidění a strojového učení. OpenCV byl vytvořen k poskytnutí základní infrastruktury pro aplikace počítačového vidění. Knihovna obsahuje přes 2500 optimalizovaných algoritmu pro práci s počítačovým viděním a strojovým učením. Obsahuje například algoritmy na rozpoznávání tváří, sledování pohybu kamery nebo získání 3D modelu ze stereo kamery (zařízení obsahující 2 kamery umístěné vedle sebe) a mnohé další. Knihovna je hojně využívána jak ve výzkumu, tak i v komerčním sektoru. Knihovna je používána dobře známými firmami jako jsou Google, Microsoft, Interl, IBM, Sony a další. Knihovnu lze použít v mnoha programovacích jazycích jako je C++, Python, Java nebo Matlab a podporuje platformy Windows, Linux, Android a MacOS. [17]

6.2 Testování jednotlivých algoritmu

Veškeré testování jsem prováděl na již zmíněných datech z MSR DailyActivity 3D datasetu. Jelikož je počet provádění dané akce velmi malý (10 záznamů každé akce), nedokáží se akce dokonale natrénovat. Abych zvýšil počet testovacích dat, u metody DTW jsem prováděl testy tak, že se vždy bral jeden záznam jako testovací a ostatních 9 záznamů pro danou akci sloužilo jako trénovacích. Jelikož si jsou některé sekvence příliš podobné a rozdíly nejsou dost výrazné na to, aby to mnou vytvořené je algoritmy dokázaly rozpoznat. Použil jsem proto pro testování

vždy menší počet akcí, než kolik celkově obsahuje datová sada. Největší rozsah testování jsem prováděl s použitím poslední představené metody, která měla v testech nejlepší výsledky.

6.2.1 Testování DTW

Celkově při testování 4 různých akcí dostáváme výsledky, které jsou zobrazeny v tabulce 1.

akce	neredukovaný počet úhlů	redukovaný počet úhlů
jedení	90%	100%
čtení	100%	50%
psaní	60%	80%
vysávání	80%	30%

Tabulka 1: Procentuální úspěšnost rozpoznávání akcí algoritmem DTW

Výsledky v tabulce 1 ukazují, že algoritmus použitý v tomto dokumentu skutečně poznává různé akce. Celkově má použití neredukovaných počtu úhlů větší přesnost, než metoda s redukovaným počtem úhlů. Lepší kvalita rozpoznávání má ale své zápory v čase, který je potřeba na rozpoznávání akce. Na porovnávání dvou akcí potřebovala metoda neredukovaných počtu úhlů o třetinu více času, než pomocí metody s redukovaným počtem úhlů. Přesnost metody by se dala navíc zlepšit odladěním thresholdů pro svolení správných úhlů, popřípadě použití předdefinovaných úhlů.

6.2.2 Testování neuronových sítí

Při testování neuronových sítí se musí postupovat trochu jinak, než u DTW. Při DTW jsme měli celý úsek záznamu a poznávali jsme to v celé jeho délce, o jaký pohyb se jedná. Pří použití neuronových sítí máme ale celý záznam rozdělený na velké množství malých segmentů.

Při testování těchto segmentů budeme brát v úvahu 2 druhy testů. V jednom případě budeme brát videozáznam jako jednu akci, a podle toho, kolik bylo správně nalezených segmentů zobrazíme. V druhém případě budeme zjišťovat celkový počet segmentů, které byly správně určeny a která špatně.

Prvně se podíváme na to, jaký vliv má velikost neuronové sítě na počet správných výsledků a časovou náročnost, jak dlouho trvá naučit danou neuronovou síť. Vyzkoušíme různé počty skrytých neuronů a podíváme se, jaký vliv na celkovou funkčnost neuronové sítě mají.

V tabulkách 2 a 3 se podíváme na srovnání jednotlivých velikostí neuronových sítí a jejich vliv na čas, kterou potřebuje neuronová síť na naučení a na přesnost výsledků. Testování je prováděna na 6 různých typech akcí, ve kterém je 5 záznamů použito jako trénovací data a dalších 5 záznamů pro každou akci data testovací. Po zpracování a rozdělení do sekvencí se dostáváme k součtu 1490 segmentů, které byly použity na naučení sítě a 5633 segmentů, které sloužily k testování.

Prvně se podíváme na metodu, která používá k rozpoznání differencí mezi počátečním a koncovým úhlem. Výsledky jsou zobrazeny v tabulce 2.

počet skrytých neuronů	úspěšnost bez aproximace	úspěšnost s aproximací	doba potřebná k naučení sítě
5	22.55%	31.35%	12.2s
10	28.85%	43.51%	15.5s
25	34.67%	40.80%	21.3s
50	38.81%	47.51%	34.0s
75	36.91%	44.93%	54.7s
100	35.38%	42.93%	1m 4.7s
250	36.30%	43.87%	2m 32.2s
500	16.71%	13.17%	5m 2.9s
1000	17.91%	12.23%	10m 41.6s

Tabulka 2: Vliv počtu skrytých neuronů na rychlost a úspěšnost rozpoznávání akcí při metodě difference počátečních a koncových úhlů

Při této metodě je počet vstupních neuronů roven počtu úhlů kostry, což je 136. Počet výstupů je stejný, jako počet akcí (6). V tabulce 2 vidíme vliv velikosti skryté vrsty na funkčnost.

Stejný způsob použijeme i na neuronovou síť používající k rozpoznávání akcí několik postojů kostry.

počet skrytých neuronů	úspěšnost bez aproximace	úspěšnost s aproximací	doba potřebná k naučení sítě
5	55.33%	59.51%	21.9s
10	54.36%	55.92%	29.6s
25	71.20%	72.25%	1m 8.7s
50	72.68%	73.67%	2m 11.6s
75	70.19%	70.18%	3m 1.7s
100	74.70%	75.38%	3m 58.7s
250	63.16%	64.16%	10m 17.2s
500	71.70%	71.26%	22m 37.9s
1000	66.03%	67.57%	50m 1.8s

Tabulka 3: Vliv počtu skrytých neuronů na rychlost a úspěšnost rozpoznávání akcí při metodě použití průběžných postojů kostry

V tabulce 3 můžeme vidět výsledky pokusu s velikostí neuronových sítí při použití metody zachytávání průběžného postoje kostry. Berme v úvahu, že počet neuronů v metodě, která po-

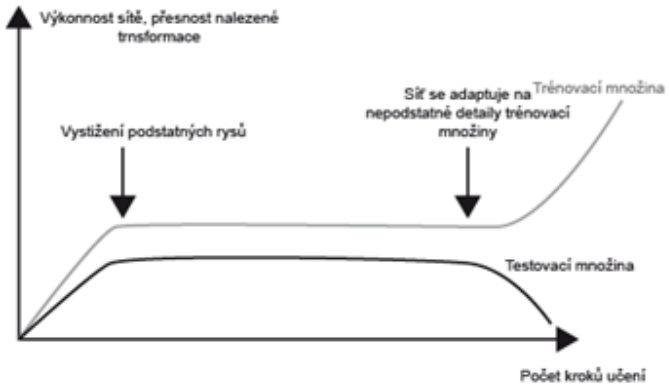
užívá několik postojů, je 544 vstupních neuronů a 6 výstupních neuronů (jelikož testujeme vliv velikost sítě pro 6 akcí). Počet synapsí můžeme spočítat jako:

$$S = Ni * Nh + Nh * No \tag{4}$$

kde Ni je počet vstupních neuronů, Nh je počet skrytých neuronů a No je počet výstupních neuronů. Každá tato synapse obsahuje váhu, kterou je nutné určit pro správné fungování sítě. Při použití 1000 skrytých neuronů se dostáváme až k počtu 550 tisíc vah, které je potřeba zjisit.

Z obou tabulek můžeme vyčíst, že do určitého počtu skrytých neuronů přesnost rozpoznávání narůstá. S tím bohužel narůstá i doba, potřebná k naučení neuronové sítě. To pokračuje do určitého bodu, kdy již větší počet skrytých neuronů nepomáhá v rozpoznávání akcí. Úroveň správnosti výsledků poté zůstává přibližně stejná nebo nižší, jen čas potřebný k naučení sítě stále roste. Při velkém počtu skrytých neuronů pak dochází k takzvanému přeučení sítě a rozpoznávací schopnost sítě klesá.

Obecně platí, že pokud síť obsahuje malý počet neuronů, její schopnost vystihnou a popsat závislosti v trénovacích datech je slabší. Pokud bude síť naopak obsahovat příliš velký počet neuronů, tato síť pravděpodobně nebude mít problém navést a reprezentovat závislosti v trénovacích datech, ale její schopnost generalizace, tedy vystihnou správný výsledek na nových datech, může být horší. Takovému jevu se říká přeučení sítě (overfitting). K přeučení může také docházet ve chvíli, kdy model obsahuje velký počet vstupních parametrů a relativně málo pozorování, což je přesně případ této práce. Cílem není maximalizace výkonu sítě na trénovacích datech, ale rozumný kompromis mezi trénovacím výkonem a schopností zevšeobecňovat znalosti i na nových datech[14].



Obrázek 30: Přeučení sítě (převzané z [14])

Na obrázku 30 je znázorněno, jak takové přeučení může vypadat. Do sítě bychom neměli přidávat žádné další skryté neurony ve chvíli, když už vystihla všechny podstatné rysy dané akce. To zaručí optimální poměr mezi rychlostí a výkonností neuronové sítě.

Při porovnání obou tabulek můžeme vidět, že metoda používající několik postojů k určení akce má mnohem lepší výsledky, než metoda používající rozdíly počátečního a koncového úhlu.

Jelikož však má mnohem více vstupních neuronů, doba potřebná k naučení sítě je několikanásobně větší.

Za zmínku stojí to, že metoda aproximace výsledků dokázala ve většině případů vylepšit výsledky, které získáváme z neuronové sítě. Zlepšení lze mnohem lépe vidět u metody s použitím diference úhlů.

Jelikož má metoda použití několika úhlů mnohem lepší výsledky, zaměříme se dále právě na ni a podíváme se podrobně na výsledky této metody.

6.2.3 Podrobné testování neuronové sítě

V této sekci se podíváme podrobněji na výsledky metody s použitím několika postojů kostry. Tato metoda měla mnohem lepší výsledky než metoda difference úhlů a proto ji budeme považovat za hlavní. Z minulého testování můžeme z tabulky 3 jsme zjistili, že neuronová síť dosahuje nejlepších výsledků při použití 100 skrytých neuronů, kde dosahuje úspěšnost rozpoznávání 75 procent. Podívejme se nyní podrobněji na to, jaké segmenty byly správně určeny a jaké ne. Tabulka 4 znázorňuje, jak byly jednotlivé segmenty určovány při použití neaproximovaných výsledků.

	pití	jedení	čtení	PC	vysávání	házení
pití	403	36	0	0	0	94
jedení	367	621	0	0	0	45
čtení	0	232	866	19	0	0
používání PC	116	0	66	948	0	40
vysávání	37	73	22	107	926	114
házení	9	0	0	48	0	444

Tabulka 4: Výsledky určení jednotlivých segmentů bez použití aproximace (Řádky znázorňují, jaké akce byly na segmentu prováděny a sloupce znázorňují, jak byl daný segment určen neuronovou sítí)

Po použití aproximace dostáváme o trochu lepší výsledky zobrazené v tabulce 5.

	pití	jedení	čtení	PC	vysávání	házení
pití	411	34	0	0	0	88
jedení	364	626	0	0	0	43
čtení	0	238	864	15	0	0
používání PC	117	0	64	950	0	39
vysávání	41	61	17	95	958	107
házení	13	0	0	51	0	437

Tabulka 5: Výsledky určení jednotlivých segmentů za použití aproximace (Řádky znázorňují, jaké akce byly na segmentu prováděny a sloupce znázorňují, jak byl daný segment určen neuronovou sítí)

Tabulky 4 a 5 znázorňují rozeznávací výsledky jednotlivých segmentů. Levá část značí, jakou akci daný segment znázorňuje a horní část znázorňuje to, jak algoritmus danou akci určil. Na diagonále jsou správné výsledky, kdy algoritmus dobře rozpoznal, o jakou akci se jedná a hodnoty, které nejsou na diagonále, jsou chybně určené segmenty. Z tabulek lze dobře rozpoznat, že algoritmus má celkem vysokou úspěšnost, pokud se jedná o odlišné pohyby. Pokud se podíváme na akce pití a jedení, jedná se o velmi podobné akce, při kterých subjekt přikládá něco k ústům. U těchto akcí vidíme, že jedení určil při více než 1/3 segmentů jako akci pití. V tabulce 6 jsou pak zobrazeny procentuální výsledky aproximované metody nad stejnou množinou.

	pití	jedení	čtení	PC	vysávání	házení
pití	77.1	6.4	0	0	0	16.5
jedení	35.2	60.6	0	0	0	4.2
čtení	0	21.3	77.4	1.3	0	0
používání PC	10	0	5.5	81.2	0	3.3
vysávání	3.2	4.8	1.3	7.4	74.9	8.4
házení	2.6	0	0	10.2	0	87.2

Tabulka 6: Procentuální rozložení segmentů bez použití aproximace (Řádky znázorňují, jaké akce byly na segmentu prováděny a sloupce znázorňují, jak byl daný segment určen neuronovou sítí)

Dosavadní testování se zaměřilo na rozpoznávání jednotlivých segmentů. Druhým krokem je testování, kdy budeme považovat celý záznam jako jednu akci. Podobně jako při DTW se vezme celý záznam a nechá se ho algoritmem rozpoznat. Při použití této metody pro stejná data, na kterých se testovaly segmenty dostáváme tabulku 7. V té lze poznat například sníženou přesnost při čtení, kdy chybně určené záznamy čtení určené jako jedení měli jen velmi malý počet segmentů. Naproti tomu správně určené záznamy obsahovaly mnohem více segmentů. Tím se může zdát, že rozpoznávací schopnost v tomto případě klesla, ale pořád se rozpoznává

stejný počet segmentů stejně, jak to bylo zobrazeno v minulých tabulkách. Naproti tomu při akci vysávání bývalo při každém provádění rozpoznáno hodně jiných šumových akcí. Ty ale nedokázaly předčít svým počtem hlavní akci a i když máme v předešlé tabulce 6 hodně malých nepřesností, ve výsledku je celková akce rozpoznána jako ta správná a algoritmus určí danou akci se 100 procentní přesností.

	pití	jedení	čtení	PC	vysávání	házení
pití	80	0	0	0	0	20
jedení	40	60	0	0	0	0
čtení	0	40	60	0	0	0
používání PC	20	0	0	80	0	0
vysávání	0	0	0	0	100	0
házení	0	0	0	20	0	80

Tabulka 7: Procentuální rozložení segmentů za použití aproximace (Řádky znázorňují, jaké akce byly na segmentu prováděny a sloupce znázorňují, jak byl daný segment určen neuronovou sítí)

Při celkovém počtu 6 akcí dostáváme celkem dobré výsledky, ale pokud se pokusíme použít stejný postup nad 15 akcemi, dostáváme mnohem horší rozpoznávací schopnost. Přece jen, mít 15 různých akcí a pro každou akci mít jen 5 učících vzorků je příliš málo na to, aby se síť správně naučila. I přesto jsem je otestoval pro všech 15 akcí. Množství segmentů při učení bylo 3215.

Než se dostaneme k podrobným výsledkům pro 15 akcí, otestujeme ještě jednou vliv velikosti počtu skrytých neuronů na rozpoznávací schopnosti algoritmu. Podíváme se, jaký vliv má zvětšení počtu akcí a učících segmentů na nejlepší počet skrytých neuronů. V tabulce 8 jsou zobrazeny výsledky tohoto testu.

počet skrytých neuronů	úspěšnost bez aproximace	úspěšnost s aproximací	doba potřebná k naučení sítě
50	41.05%	42.44%	5m 0.6s
100	47.81%	49.44%	9m 44.8s
250	41.47%	42.17%	24m 12s

Tabulka 8: Vliv počtu skrytých neuronů na rychlost naučení a schopnost rozpoznávání akcí neuronou sítí při použití většího množství akcí

I v tomto případě vyšla nejoptimálnější velikost sítě se 100 neurony ve skryté vrstvě. Proto použijeme toto nastavení sítě na podrobné zobrazení výsledků pro 15 akcí. Aproximované procentuální výsledky jsou zobrazené v tabulce na obrázku 31.

	Pití	Jedení	Čtení	Telefonování	Psaní	Používání PC	Vysávání	Oslavování	Nicnedělání	Házení	Hraní her	Lehnutí	Hraní na kytaru	Posatvení se	Sednutí si
Pití	55,3	9,2	0	35,5	0	0	0	0	0	0	0	0	0	0	0
Jedení	8,7	67	0	22,4	0	0	0	0	0	0	0	0	0	0	1,9
Čtení	0	17,9	63,9	0	0,7	16,8	0	0	0	0	0	0,7	0	0	0
Telefonování	33,8	9,5	0	45,9	5,8	0	0	0	3,7	0	0	1,3	0	0	0
Psaní	5,4	2	51,9	0,8	26,2	2,5	0	0	6,2	0	3,6	1,4	0	0	0
Používání PC	2,6	0	1,2	14,4	0	62,1	0	0	0	1	0,9	0,4	0	0	17,4
Vysávání	0	0,5	0	7,6	5,7	7,3	44,9	2,7	10	1,2	0,8	8,7	2,5	5,3	2,8
Oslavování	0	0	0	1,7	0	0	0	90,5	2,1	1,3	0	0,8	0	2,7	0,9
Nicnedělání	22,7	0	0	7,2	0	0	0	0	45,5	0	0	0	0	0	24,6
Házení	4,6	1,8	0	12,6	0	10,2	0	3	0	39,3	0	21,5	0	0	7
Hraní her	0	27	9,8	0	7,5	0	0	11,5	0	0	40,6	0	0	0	3,6
Lehnutí	1,5	0	0	0	0,5	0	5,8	0	13,8	0	0	49,9	0	4	24,5
Hraní na kytaru	6,8	3,2	0	14	10,2	17,9	6	1,8	4	0,1	2	2,6	19,9	0	11,5
Posatvení se	0	0	0	8,4	0	0	0	0	11,2	0	0	0	0	48,7	31,7
Sednutí si	0	0	0	0	0	0	0	0	24,5	0	0	16,1	0	21,2	38,2

Obrázek 31: Procentuální rozložení segmentů při rozpoznávání 15 akcí neuronovou sítí (Řádky znázorňují, jaké akce byly na segmentu prováděny a sloupce znázorňují, jak daný daný segment určen neuronovou sítí. Na diagonále je znázorněna procentuální úspěšnost)

Na obrázku 31 můžeme vidět, že oproti testům s menším počtem akcí dochází ke zvýšenému počtu chyb. Zde si již algoritmus trochu neví rady, ale stále rozpozná správné výsledky se skoro 50 procentní úspěšností. Na důvody, proč vznikají takovéto chyby, se podíváme v další sekci.

6.3 Důvody chybného rozpoznání

Jak můžeme z jednotlivých výsledků algoritmů, jak u DTW, tak u postupů s použitím neuronových sítí dochází k chybným rozpoznáním. Důvodů je hned několik.

6.3.1 Nedokonalost trénovacích a testovacích dat

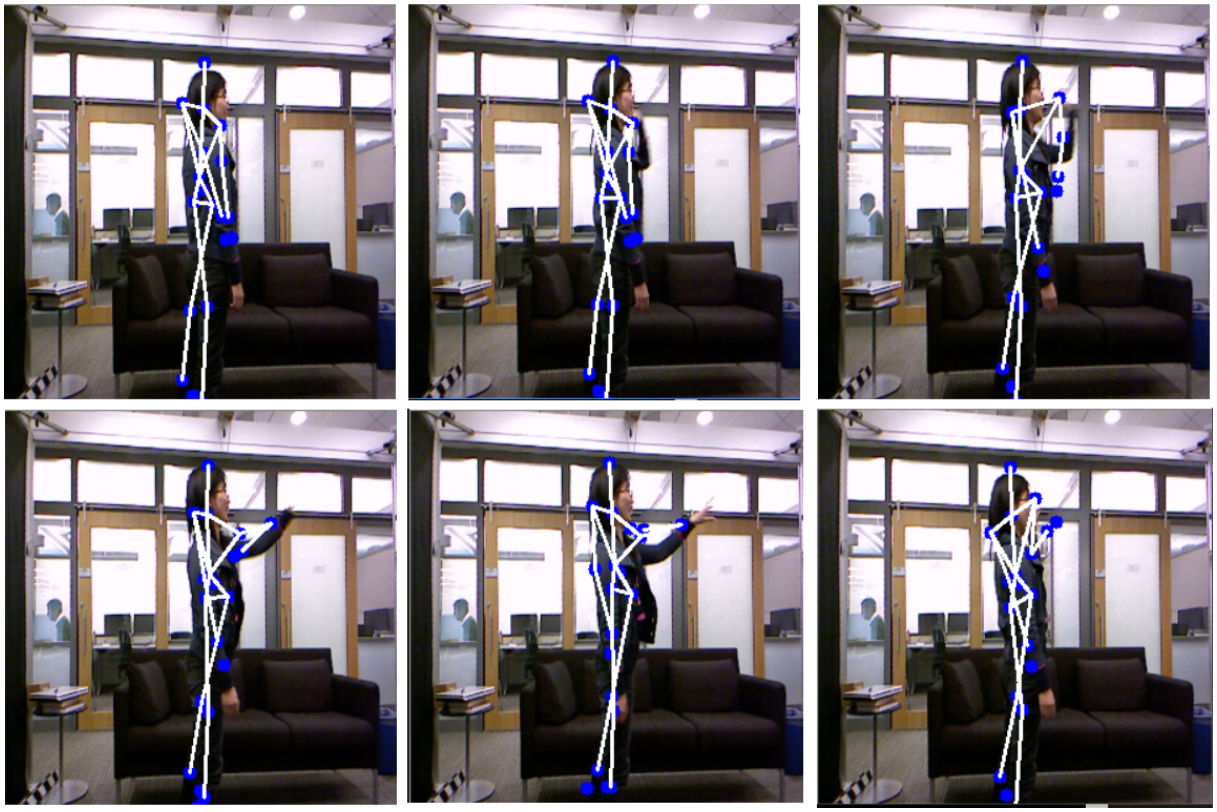
V první řadě musíme brát ohled nedokonalosti jak trénovacích, tak i testovacích dat. Jednotlivé záznamy jsou o trochu delší, než by bylo potřeba a člověk na nich dělá víc než jen akci potřebnou k rozpoznávání. Jednotlivé subjekty před, po, ale i během akce provádějí pohyby, které pro danou akci nejsou určující, ale spíš patří k úplně jiné akci. Například pro záznamy, kdy subjekt hází míčkem, patří více než polovina snímků k jinému pohybu. To znemožňuje správné rozpoznávání akcí. Řešením by bylo v první řadě ořezat každý záznam jen na oblast, kdy se opravdu pohyb provádí. Dalším krokem by bylo znásobit počet trénovacích dat, protože jen 10 záznamů pro daný pohyb je příliš málo. Velké komerční neuronové sítě používají k trénování stovky tisíc, milióny nebo i víc záznamů. Existují velké trénovací sady pro naučení rozpoznávání, co se nachází na

obrázku. V době psaní této práce neexistuje žádná datová sada, která by obsahovala velké množství sekvencí, které by se daly v této práci použít. Již však sada MSR DailyAction 3D, která obsahuje 320 velmi krátkých sekvencí (řádově desítky až pár stovek snímků) zabere na disku přes 25GB místa.

6.3.2 Chybně určená pozice kostry člověka

Druhým problémem jsou už zmíněné nedokonalosti pozice bodů kostry. Buď je to způsobeno tím, že člověk není kamerou správně zachycen, protože například některé jeho části jsou pro ní skryté, nebo dochází k nedokonalosti algoritmu. Důvodů může být více a můžou se kombinovat, takže výsledná kostra nekoresponduje s postavením člověka. Jelikož je počet záznamů malý, každá takováto odchylka výrazně ovlivňuje rozpoznávací schopnosti algoritmů.

Na obrázku 32 můžeme vidět příklad špatně rozpoznané kostry při akci házení. Hlavním důvodem v tomto případě hraje roli to, že subjekt je ke kameře natočen tak, že ty nejdůležitější části těla, které provádějí daný pohyb, jsou pro kameru skryty.



Obrázek 32: Příklad špatně rozpoznané kostry

Na obrázku 32 lze vidět, že levou ruku člověka buď algoritmus nerozpozná vůbec, nebo ji určí chybně. Když už měla kamera možnost vidět danou ruku snímaného člověka, určila ji špatně. Na některých snímcích ze sekvence můžeme vidět, že bod lokte je v oblasti, kde se nachází dlaň

a bod kostry v oblasti dlaně je umístěn v oblasti ramene člověka. Celkově je kostra deformovaná a jen velmi těžko by se dalo poznat, o jakou akci se jedná.

6.3.3 Různé možnosti provádění stejné akce

Třetím problémem, který stále souvisí s malým množstvím dat je ten, že subjekty na sekvencích neprovádí akci stejně. Každý používá jiné části ke svému pohybu a navíc v mnoha sekvencích jsou úmyslně stejné akce prováděny úplně jiným způsobem. Vezměme si příklad házení. Na sekvenci obrázků 33 vidíme, že daný člověk provádí akci házením míčku tak, že zvedne ruku nad hlavu a hodí. Dobrá, to je jednoduše rozpoznatelné, pokud udělá druhý člověk podobný pohyb.

Obrázek 33: Příklad akce házení

Podívejme se ale na druhou sekvence obrázků 34, která je také popsána jako házení míčku. V tomto případě má daný subjekt při provádění akce úplně jiný postoj, ale stále hází míček přibližně v oblasti hlavy. S tímto už začíná mít algoritmus drobné problémy, nachází se zde pár chybných určení, ale stále z větší části rozpozná, že se jedná o akci házení.

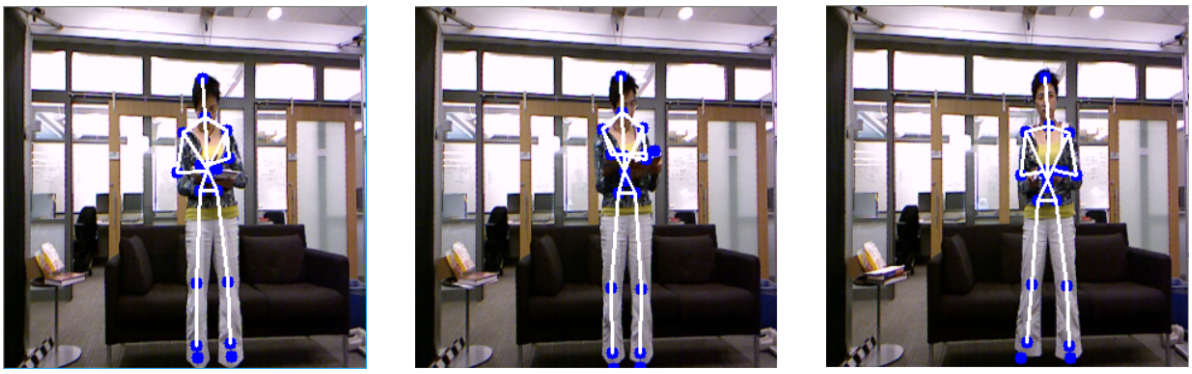
Obrázek 34: Házení při jiném postoji subjektu

Ve třetím případě všechny algoritmu zcela selžou. Na sekvenci obrázků 35 máme subjekt, který se rozhodl házet míčkem od boku. Jelikož je trénovací množina velmi malá a žádný podobný pohyb takový, ve kterém by uživatel házel míčkem od boku nebyl proveden, rozpoznání zcela selže a dostáváme náhodné hodnoty. Toto se dá vyřešit rozšířením trénovací množiny o vzorky zahrnující i tyto pohyby. Trénovací data použitá v této práci nejsou dost obsáhlá na to, aby se tomuto pohybu přizpůsobila.

Obrázek 35: Jiný způsob házení

6.3.4 Příliš velká podobnost akcí

Dalším z řady problémů je i přílišná podobnost akcí. Vezměme si například takové pohyby jako jsou čtení, psaní nebo hraní na herním ovladači. Na obrázku 36 jsou zobrazeny tyto 3 různé akce prováděné různými lidmi.



(a) Psaní (b) Čtení (c) Používání herního ovladače

Obrázek 36: Ukázka podobných postojů při různých akcích

Jelikož nejsou rozdíly mezi pozicemi bodu v oblasti dlaně a zápěstí dostatečně daleko od sebe a algoritmus v těchto částech spíše hádá body kostry, než aby správně určovala přesně tyto body. V této aplikace se vždy používá jen jeden z těchto bodů. I kdybychom je ovšem zahrnuli, výsledek by to výrazně nezměnilo. Na ukázce různých akcí na obrázcích 36 jsou si postoje příliš podobné a algoritmus rozpoznává tyto pohyby podle celkového postoje. To způsobí, že jsou tyto pohyby velmi jednoduše zaměnitelné. Abychom mohli rozpoznat i tyto miniaturní rozdíly akcí, museli bychom mít velmi přesná data o pozicích více částí těla, než ty, co používáme v této chvíli.

Celkově je to jen další problém, který zmenšuje rozpoznávací schopnosti zde propagovaných algoritmů.

7 Závěr

Témetem této diplomové práce bylo prozkoumat metody používané pro rozpoznávaní lidských akcích. Následně se zaměřit na některou z těchto metod, naimplementovat a následně otestovat.

V této práci jsem se zaměřil na 3 různé způsoby, které lze použít na rozpoznání akcí. První metoda pomocí DTW se zaměřila na rozpoznávaní akcí v uložených sekvencích. Výhodou této metody byla rychlost, s jakou se daly akce naučit. Na druhou stranu byla velmi pomalá, když se jednalo rozpoznávání. Pro rozpoznávání v reálném čase jsem testoval 2 různé metody, které byly založeny na neuronových sítích. Jedna rozpoznávala rozdíl mezi postojem na začátku a na konci krátkého segmentu a druhá uchovávala průběžné postoje v těchto segmentech. Tyto metody jsem vylepšil aproximační funkcí, která dokázala omezit škodlivé vlivy výkyvů rozpoznaného pohybu. Všechny metody mají stejný základ v tom, že používají pro reprezentaci člověka jeho kostru.

Všechny tyto metody jsem implementoval v jazyce C++ pomocí knihovny OpenCV. Testování bylo prováděna na testovací sadě MSR DailyAction 3D, která obsahovala 16 různých akcí prováděné vždy 10 lidmi. Data obsahovala jak videozáznam, tak i záznam hloubkové mapy a důležité body kostry člověka. Záznamy byly pořízeny zařízením Kinect. Testování hodně utrpělo nedostatečným rozsahem těchto dat, protože pro správné naučení neuronové sítě je potřeba velká trénovací sada. Nejlépe byla vyhodnocena metoda s použitím neuronových sítí při průběžném uchovávání postoje člověka. Při 6 rozpoznávaných akcích a 5 trénovacích vzorků pro každou akci jsme dosáhli na úspěšnost více než 75%. Při testování 15 akcí a stejném počtu trénovacích vzorů pro každou akci klesla úspěšnost přibližně na 50%. Podrobné výsledky a důvody, proč je taková úspěšnost jsem podrobně popsal v sekci testování v kapitole 6.

Pro vylepšení algoritmu by bylo v první řadě nutné zvýšit velikost trénovacích dat. Druhým nejvýznamnějším přínosem by bylo začít brát ohled na to, co se děje ve videozáznamu snímajícím viditelné spektrum světla. Výhodou by bylo použití algoritmů, které by dokázaly rozpoznat, jaké objekty se v obraze nacházejí a s čím daný člověk provádějící akci interaguje (například jestli drží hrnek nebo telefon). Při správné kombinaci použití jak modelu kostry, tak i detekce objektů, by se dokázala úspěšnost rozpoznávání akcí značně zvýšit.

Pro vylepšení rozpoznávaní a rychlostí sítí by mohla být použitá složitější konstrukce, která by více vyhovovala řešení daného problému. Navíc by se daly použít technologie, které dokáží využít lépe výkon počítače. Místo jednoho jádra procesoru by mohlo pracovat tisíce jader grafických karet a rychlost by se zvýšila o pár řádů. To by dovolilo používat o mnoho komplexnější sítě.

Celkově je rozpoznání lidských akcí velice zajímavou oblastí počítačového vidění, která se bude se zlepšujícími se technologiemi a algoritmy stále více a více zpřesňovat. Využití těchto systémů může být například v hlídání budov, městských systémech, kde bychom mohli například detekovat vznikající nepokoje (rvačky a podobně).

Literatura

[1] SCHULDT, Christian; LAPTEV, Ivan; CAPUTO, Barbara. Recognizing human actions: A local SVM approach. In: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on. IEEE, 2004. p. 32-36.

[2] LAPTEV, Ivan, et al. Learning realistic human actions from movies. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE, 2008. p. 1-8.

[3] BOBICK, Aaron F.. ; DAVIS, James W.. . The recognition of human movement using temporal templates. IEEE Transactions on pattern analysis and machine intelligence, 2001, 23.3: 257-267.

[4] XIA, Lu; CHEN, Chia-Chih; AGGARWAL, J. K. View invariant human action recognition using histograms of 3d joints. In: Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on. IEEE, 2012. p. 20-27.

[5] JI, Shuiwang, et al. 3D convolutional neural networks for human action recognition. IEEE transactions on pattern analysis and machine intelligence, 2013, 35.1: 221-231.

[6] CHEN, Chen; LIU, Kui; KEHTARNAVAZ, Nasser. Real-time human action recognition based on depth motion maps. Journal of real-time image processing, 2016, 12.1: 155-163.

[7] CHEN, Chen; JAFARI, Roozbeh; KEHTARNAVAZ, Nasser. Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In: Image Processing (ICIP), 2015 IEEE International Conference on. IEEE, 2015. p. 168-172.

[8] YANG, Xiaodong; ZHANG, Chenyang; TIAN, YingLi. Recognizing actions using depth motion maps-based histograms of oriented gradients. In: Proceedings of the 20th ACM international conference on Multimedia. ACM, 2012. p. 1057-1060.

[9] WANG, Jiang, et al. Robust 3d action recognition with random occupancy patterns. In: Computer vision–ECCV 2012. Springer Berlin Heidelberg, 2012. p. 872-885.

[10] VEMULAPALLI, Raviteja; ARRATE, Felipe; CHELLAPPA, Rama. Human action recognition by representing 3d skeletons as points in a lie group. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2014. p. 588-595.

[11] SEMPENA, Samsu; MAULIDEVI, Nur Ulfa; ARYAN, Peb Ruswono. Human action recognition using dynamic time warping. In: Electrical Engineering and Informatics (ICEEI), 2011 International Conference on. IEEE, 2011. p. 1-5.

[12] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, Real-Time Human Pose Recognition in Parts from a Single Depth Image, in CVPR, IEEE, June 2011

[13] Christos Stergiou and Dimitrios Siganos [Online]. http://www.doc.ic.ac.uk/ nd/surprise_96/journal/vol4/cs11/ report.html

[14] Úvod do neuronových sítí. StatSoft [online]. Praha: StatSoft CR, 2013 [cit. 2017-04-10]. Dostupné z: http://www.statsoft.cz/file1/PDF/newsletter/2013_02_05_StatSoft_Neuronove_site_linky.pdf

[15] Neural Networks. The nature of code: simulating natural systems with processing [online]. Version 1.0, generated December 6,2016. New York: Free Software Foundation, 2012 [cit. 2017-04-10]. ISBN 9780985930806.

[16] XIA, Lu; CHEN, Chia-Chih; AGGARWAL, J. K. View invariant human action recognition using histograms of 3d joints. In: Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on. IEEE, 2012. p. 20-27.

[17] Neural Networks. OpenCV docs [online]. 2017 [cit. 2017-04-10]. Dostupné z: http://docs.opencv.org/2.4/modules/ml/doc/neural_networks.html

[18] Object Classification using feature extraction and bag of features (applications in OpenCV). Unraveled thinking... [online]. 2017 [cit. 2017-04-10]. Dostupné z: http://unraveledthinking.blogspot.cz/2012/11/object-classification-using-feature.html

[19] Inside the Newest Kinect for Windows SDK—Infrared Control. Microsoft [online]. 2017 [cit. 2017-04-10]. Dostupné z: https://blogs.msdn.microsoft.com/kinectforwindows/2012/12/07/inside-the-newest-kinect-for-windows-sdkinfrared-control/

[20] Mocap [online]. 2017 [cit. 2017-04-10]. Dostupné z: http://http://motion-capture.blogspot.cz/

[21] Mocap suit. iPi Soft [online]. 2017 [cit. 2017-04-10]. Dostupné z: http://ipisoft.com/community/customer-highlights/

[22] Matematický model a aktivní dynamika neuronu. Matematická bilogie [online]. Brno: Institut biostatistiky a analýz Masarykovy univerzity [cit. 2017-04-10]. Dostupné z: http://portal.matematickabiologie.cz/index.php?pg=analyza-a-hodnoceni-biologickych-dat–umela-inteligence–neuronove-site-jednotlivy-neuron–jednotlivy-neuron–matematicky-model-a-aktivni-dynamika-neuronu

[23] DTW algorithm [Online]. Jiang Wang [cit. 2017-04-10]. Dostupné z: http://users.eecs.northwestern.edu/ jwa368/my_data.html

[24] Datasets [Online]. Computational Biology [cit. 2017-04-10]. https://www.psb.ugent.be/cbd/papers/gentxwarper/DTWalgorithm.htm

[25] CASSISI, Carmelo, et al. Similarity measures and dimensionality reduction techniques for time series data mining. In: Advances in data mining knowledge discovery and applications. Intech, 2012.

A Příloha na DVD

- ReadMe obsahující základní pokyny pro ovládání testovací aplikace
- Testovací aplikace (+ přiložená testovací data)
- Projekt testovací aplikace pro Visual Studio 2013
- PDF obsahující text diplomové práce